LV-Chat: Facilitating Long Video Comprehension

Anonymous ACL submission

Abstract

Enabling large language models (LLMs) to read videos is vital for multimodal LLMs. Existing works show promise on short videos whereas long video (longer than e.g. 1 minute) comprehension remains challenging. The major problem lies in the over-compression of videos, i.e., the encoded video representations are not enough to represent the whole video. To address this issue, we propose Long Video Chat (LV-Chat), where Frame-Scalable Encoding (FSE) is introduced to dynamically adjust the number of embeddings in alignment with the duration of the video to ensure long videos are not overly compressed into a few embeddings. To deal with long videos whose length is beyond videos seen during training, we propose Interleaved Frame Encoding (IFE), repeating positional embedding and interleaving multiple groups of videos to enable long video input, avoiding performance degradation due to overly long videos. Experimental results show that LV-Chat significantly outperforms existing methods by up to 27% in accuracy on long-video QA datasets and long-video captioning benchmarks. Codes and data will be released upon publishing.

1 Introduction

Recent works have been proposed to enhance the multimodal capabilities of large language models, extending their power beyond text to other data modalities such as images (Touvron et al., 2021; Bao et al., 2021; He et al., 2022) and audio (Hassid et al., 2023; Borsos et al., 2023; Sicherman and Adi, 2023). Among them, videos offer a unique medium through how humans perceive the real world (Li et al., 2023). To leverage this, recent efforts on augmenting LLMs' video comprehension have focused on finetuning LLMs with video instruction data such as VideoChat (Li et al., 2023), VideoChat-GPT (Maaz et al., 2023), VideoLlama (Zhang et al., 2023).

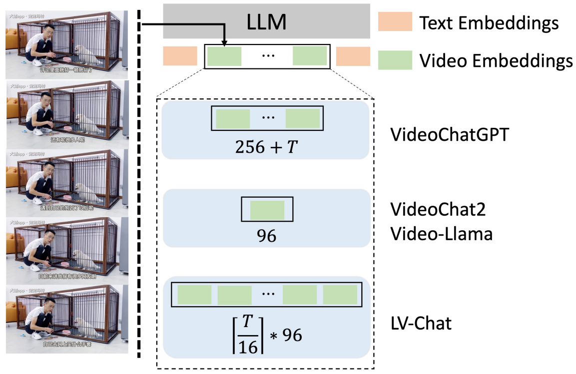


Figure 1: Previous video language models may suffer from over-compression for long video modeling (e.g., $T > 60s$) since a limited number of video tokens are used in LMs. In contrast, LV-Chat demonstrates superior performance on long videos by modeling more video tokens.

While previous video-language models have demonstrated promising results, particularly with short videos, their performance on videos *longer than one-minute* is observed to be challenging (Li et al., 2023). We believe (and empirically prove it in our experiments) that *the inability to comprehend long videos comes from the over-compression of video content.* For example, VideoChatGPT (Maaz et al., 2023) models a video of T seconds by sampling (F frames). These frames, along with a prefix of 256 tokens designated for global information, are then compressed into a total of $256 + F$ tokens. This compression strategy is insufficient for longer videos, where the complexity and information density exceed the representational capacity of the allocated tokens. On the other hand, mainly focusing on short videos, VideoChat (Li et al., 2023) and Video-Llama (Zhang et al., 2023) convert F_s sampled frames into a fixed tiny number of embeddings (96 embeddings), regardless of the video's duration, resulting in inadequate information for effective long-video representation.

In this work, we focus on the long video understanding scenario and propose a novel video language model LV-CHAT. LV-CHAT has two key components: *Frame Scalable Encoding (FSE)* and *Interleaved Frame Encoding (IFE)*. To tackle the over-compression problems, we design FSE, a new feature extraction strategy that scales the number of tokens with the video length T. Specifically, every 16 frames are compressed into 96 tokens to ensure the video information is mostly maintained during the mapping. The model is then fine-tuned on these compressed $\lceil \frac{T}{16} \rceil * 96$ embeddings. To overcome the out-of-distribution (OOD) problem encountered during inference when videos are longer than those seen during training, we introduce IFE, a novel interleaving strategy to repeat positional embeddings and interleave multiple groups of videos to enable long video input and avoid OOD issue.

We evaluate LV-CHAT in the tasks of long-video question answering (QA) and long-video captioning. We observe that existing video benchmarks (Li et al., 2024) primarily annotate a short clip of the entire video where the ground truth label is located (with such annotation, previous works only input the clip instead of the entire video). Since such annotation requires human effort to locate the answer, in our work, we investigate a more practical setup where such timestamp annotation is not available. To this end, we develop a long-video QA benchmark by randomly concatenating real video segments in MVBench (Li et al., 2024) with distractor videos, along with a long-video captioning dataset TACoS (Rohrbach et al., 2014) where we manually create the ground truth captions according to human-annotated subtitles for its long videos. We also test LV-CHAT on EgoSchema (Mangalam et al., 2023), a challenging long-video QA benchmark. The experimental results show that LV-CHAT largely improves the accuracy over baselines in our curated long-video QA task (600s) even with FSE only (up to 21% improvement in accuracy) and adding IFE further improves accuracy (up to 27%), highlighting the potential of LV-CHAT and shedding light on future advancements in long-video language models.

2 Related Work

2.1 Long Context Modeling

There are lots of long context modeling techniques, among which modifying positional embeddings resembles our method the most. The most similar one to Interleaved Frame Encoding is Self-Exntending LLMs (Jin et al., 2024). Other works include adopting relative positional embedding (Press et al., 2021), positional interpolation (Chen et al., 2023) and positional extrapolation (Sun et al., 2023). As mainly focus on text domain, these works are orthogonal to our use cases with multimodality data.

2.2 Video Question Answering

Video Question Answering (VideoQA) has been a popular task for evaluating the model's ability to understand videos. Typical works pretrain a video-text model and perform a successive fine-tuning on VideoQA (Zellers et al., 2021; Bain et al., 2021; Miech et al., 2019; Wang et al., 2022; Fu et al., 2021; Zeng et al., 2022; Li et al., 2022). These works are focused specifically on video question answering, and large language models are not introduced here, which might limit the interpretation of the video content and the application of the model.

2.3 Enabling LLMs to Process Videos through Descriptive Textualization

A foundational approach towards equipping LLMs with video understanding capabilities involves the extraction of information from each frame of the video, subsequently converting this data into a textual format for LLM processing. Notable implementations of this strategy include ChatVideo (Wang et al., 2023a) and VideoChat-Text (Li et al., 2023). These methods are limited by their reliance on textual conversion, which might pose problems when there are scenes beyond text descriptions.

2.4 Enabling LLMs to Process Videos via Adapters

An emergent trend in recent research focuses on introducing adapters to bridge the gap between visual representations and the textual embedding space of LLMs. Some works take the first step on images such as VC-GPT (Luo et al., 2022), VisualGPT (Chen et al., 2022), Mini-GPT4 (Zhu et al., 2023) and LlaVa (Liu et al., 2023), which proposes the adapters to map the visual encoder outputs into the word embedding space, enabling direct processing of image data with LLMs. Based on these models with image understanding capabilities, VideoChat-Embed and VideoChat2 (Li et al., 2023) propose to encode the videos into the embeddings with an extra adapter, where the visual encoder and the adapter is trained using video instruction datasets. Similarly, VideoChatGPT (Maaz et al., 2023) initializes from LlaVa and is trained on another comprehensive video instruction dataset. Video-Llama (Zhang et al., 2023) add audio modality into the instruction finetuning, enabling LLM to both see and hear. FrozenBiLM (Yang et al., 2022) adapts a pre-trained BiLM to multi-modal inputs and introduces a set of additional modules including adapters, which are trained on video-text data. These methods show promising results in terms of short video understanding, but they may struggle with managing long videos (typically longer than 1min). Our work is built based on the backbone VideoChat2 (Li et al., 2023) but with additional fine-tuning and design of FSE and IFE, improving long video understanding.

3 Method

3.1 Preliminary

Following VideoChat2 (Li et al., 2023), assume we are generating captions for a given video $\mathbf{V} = [\mathbf{I}_i]_{i=1,2,\cdots,F}$, where F is the total frames of the video, with $\mathbf{I}_i$ being the i-th frame. Then we need to convert the video $\mathbf{V}$ into embeddings $\mathbf{E}$:

$$\mathbf{E} = f_{vid}(\mathbf{V}). \tag{1}$$

Here f_{vid} denotes the video encoding model. Since we aim to enable the LLM to understand the video, $\mathbf{E}$ is usually trained to align the distribution of word embeddings in the LLMs f_{llm}. The next word is predicted following the equation:

$$\mathbf{P} = f_{llm}(\mathbf{E}, \mathbf{W}_{\leq t}), \tag{2}$$

where $\mathbf{W}_{\leq t}$ is the word embeddings of previous words generated in the sentence and $\mathbf{P}$ is the next-word probability distribution over the vocabulary. For the instantiation of the visual encoder f_{vid} and the large language model f_{llm}, we follow VideoChat2 (Li et al., 2023), where UMT-L (Liu et al., 2022) followed a pretrained QFormer and an extra linear adapter is used as f_{vid} to map the video frames into embeddings $\mathbf{E}$ (in the space of the word embeddings) and Vicuna-7B-v1.0 (Chiang et al., 2023) is used as f_{llm}.

3.2 Frame-Scalable Encoding

We observe a limitation in previous approaches that potential over-compression on given (long) videos can happen. To address this issue, we propose a novel encoding strategy, Frame-Scalable Encoding (FSE), based on the intuition that the number of embeddings allocated for video representation should be sufficient to cover the information within the video. The framework is shown in Figure 2. Specifically, given a long video $\mathbf{V}$, FSE requires the video to be segmented into a series of clips $\mathbf{V}_1, \cdots, \mathbf{V}_n$, each bounded by a predefined maximum frame count. Then each clip is converted into a designated number of embeddings with Eq.(1):

$$\mathbf{E}_1, \cdots, \mathbf{E}_n = f_{vid}(\mathbf{V}_1), \cdots, f_{vid}(\mathbf{V}_n) \tag{3}$$

Here each embedding $\mathbf{E}_i \in \mathbb{R}^{N \times d}, i \in \{1, \cdots, n\}$, where d is the hidden dimension of the LLM. Then we concatenate all embeddings $\mathbf{E}_1, \cdots, \mathbf{E}_n$ into the final representation $\mathbf{E}_{FSE} \in \mathbb{R}^{(n*N) \times d}$. Thus the representation $\mathbf{E}_{FSE}$ comprises $n * N$ embeddings. When the video gets longer, we could obtain more clips (n would increase), leading to more embeddings and effectively mitigating the risk of over-compression. To determine how many clips we need, we propose the following equation:

$$n = \lceil T/K \rceil \tag{4}$$

where T denotes the video's duration (measured in seconds), ensuring a minimum of one frame per second is utilized. As the backbone model VideoChat2 is trained with the embeddings $\mathbf{E} \in \mathbb{R}^{N \times d}$ (i.e., only N embeddings), it may struggle with our embeddings $\mathbf{E}_{FSE}$ which comprises $n * N$ embeddings. Thus we fine-tune the backbone model with the FSE embeddings. During training, due to the limitation of the resources and the constraint of maximal positional embeddings, we specify a maximum number of clips n_m and only sample n_m clips when videos get long, resulting in $n_m * K$ frames. During inference, for longer videos, we can keep the strategy from training, i.e., only sample n_m clips. However, we propose a more optimal solution and explain mode details in the subsequent section.

3.3 Interleaved Frame Encoding

Although Frame-Scale Encoding (FSE) could mitigate the over-compression to some extent, it may introduce another challenge: when FSE is applied to excessively long videos, we may obtain an unwieldy number of embeddings from Eq.(3). When there are overly many embeddings, it may surpass the maximum positional embeddings of the LLM. It may also encounter a problem that the embeddings during the inference is longer than the

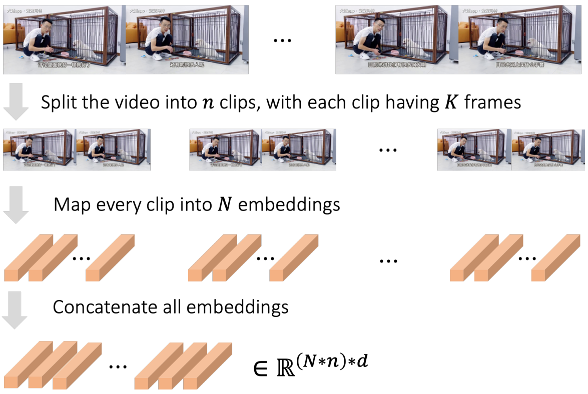


Figure 2: Illustration of Frame-Scalable Encoding. The process begins by segmenting the video into several clips. Subsequently, each clip is transformed into a set of N embeddings. These embeddings are then concatenated sequentially, forming a comprehensive input stream for the Large Language Model (LLM).

embeddings seen during training, leading to out-of-distrubition (OOD) problems. As discussed in § 3.2, a suboptimal solution could be limiting the clip numbers to be less than n_m, but this approach may still suffer from over-compression identified in § 1 as the number of embeddings is not scalable w.r.t. the video length, thereby limiting the effectiveness of FSE.

To tackle this challenge, we propose **Interleaved Frame Encoding (IFE)**. IFE employs a repetition factor, γ for the positional embeddings. Therefore, the positional embeddings are repeated at a predefined interval, γ, so that the sampled embeddings are within the range of training length, mitigating the OOD issues or potential risk of surpassing maximum positional embeddings of the LLM. The process of IFE is depicted in Figure 3. As shown in the figure, we split the video into γ groups $\mathbf{V}_1, \cdots, \mathbf{V}_\gamma$. Each group is converted into embeddings $\mathbf{E}_{FSE,1}, \cdots, \mathbf{E}_{FSE,\gamma}$ using FSE techniques. These embeddings are fed into the LLM with the same positional embeddings applied to each group. One property we wish to include is that even when one group of the video is processed in isolation, without interleaving, IFE should align with using FSE with n_m clips. To achieve this, the video is divided into γ groups in an interleaved way (shown in the Figure 3). Then each group is encoded into embeddings independently. After this encoding phase, all embeddings are interleaved before being fed into the LLM. As illustrated, maintaining only one group (e.g., removing the right part in Figure 3) effectively simulates the FSE scenario, sampling only the frames in one group (green frames in the example). Incorporating additional groups is intuitively expected to enhance the understanding of the video.

For IFE, we determine the interleaving factor γ by the following equation:

$$\gamma = \lceil \lceil T/K \rceil / n_m \rceil. \tag{5}$$

The intuition behind Eq.(5) is to make sure the number of clips in each group is less than n_m while maintaining the total amount of frames sampled could cover the whole video. Then we could sample F_s frames from the given video:

$$F_s = \lceil \lceil T/K \rceil / \gamma \rceil * \gamma * K \tag{6}$$

Thus the number of the clips would be:

$$n_i = \lceil \lceil T/K \rceil / \gamma \rceil * \gamma \tag{7}$$

With this strategy, we could sample F_s frames from the video which could cover the whole video, as a result to have more than one frame per second, ensuring effective representation of the video.

4 Experiments

4.1 Implementation Details

We initialize our model from VideoChat2 (Li et al., 2023). We set the learning rate as 2e-6, with warmup epochs=0.3, num_epochs=1, scheduler=*cos*, optimizer=AdamW. The fine-tuning is performed on 4 NVIDIA-RTX-A6000 GPUs. For FSE, we finetune our model on the instruction dataset collected for training VideoChat2 (Li et al., 2023) with the detailed datasets shown in Appendix §B.1.

For LV-CHAT, we use Eq.(4) to determine the number of frames to sample, and encode every K frames into N embeddings, where $K = 16, N = 96$. During the training, we specify $n_m = 10$. Thus if the video length T is shorter than $n_m * K = 160$, we do not need IFE and only FSE is turned on, whereas if the video length T is longer than 160, we determine the interleaving factor γ with Eq.(5) and then perform the IFE process.

4.2 Experimental Setups

We compare LV-CHAT with the following models: **VideoChat2** (Li et al., 2023): The backbone of our model without FSE and IFE. We follow the implementation[1] in VideoChat2 and sample 16 frames

[1] https://github.com/OpenGVLab/Ask-Anything/blob/main/video_chat2/mvbench.ipynb

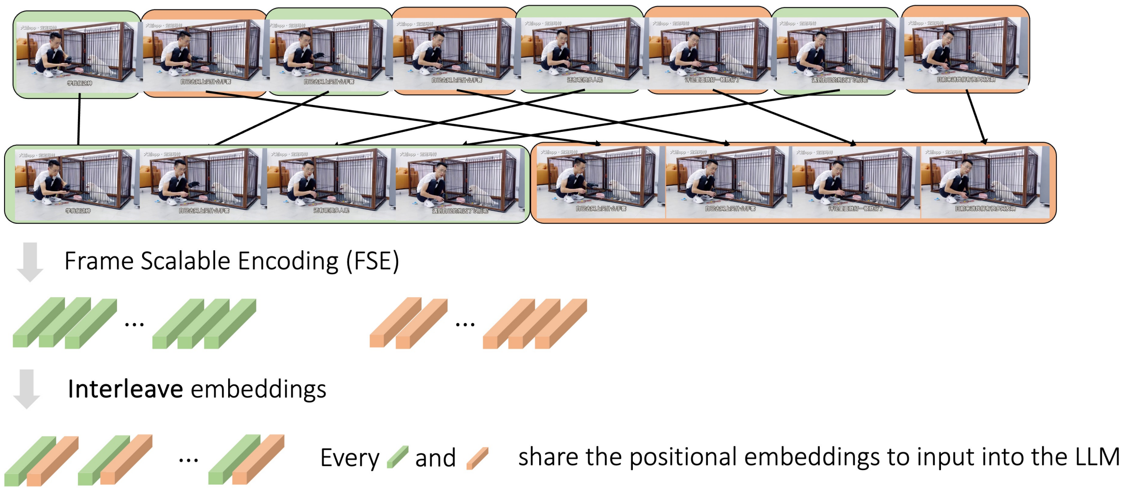


Figure 3: Illustration of Interleaved Frame Encoding (IFE). We show the example with interleaving factor γ being two. We first split the whole video into γ groups. Then we convert each part into embeddings separately. With all the embeddings, we interleave them with every γ embeddings sharing the same positional embedding.

from the given video regardless of the video length. **Video-Llama** (Zhang et al., 2023): We exclude the audio modality here for fair comparison. Following the setting from the original implementation[2] model, we use the Video-LLaMA-2-7B-Finetuned checkpoint and sample 16 frames from each video. **Video-ChatGPT** (Maaz et al., 2023): We use the same setup as in the official demo[3] and samples 100 frames from each video.

For the benchmarks, we adopt MVBench (Li et al., 2024) and extend the videos with the Street-Scene (Ramachandra and Jones, 2020) dataset to specific lengths (see §B.3). We select the following 4 out of 20 test sets from MVBench: Action Sequence(AS), Action Prediction(AP), Unexpected Action(UA), Object Interaction(OI), following the criteria detailed in Appendix§ B.2. The average length of the videos from these subsets is 25.5 seconds. We extend the original videos to 100s, 300s, and 600s respectively. All models are evaluated under the same protocol proposed in MVBench. The prompts used are summarized in Appendix §B.5. We also report the performance on all subsets of MVBench in Appendix §C.1.

4.3 Overall Performance Comparison

We report the overall performance comparison in Table 1. From this table, we can observe that LV-CHAT outperforms the previous methods significantly on almost all datasets and in almost all settings. The results demonstrate that LV-CHAT could better extract the important information from the video even when the video becomes as long as 600s. We also summarize the average results of all datasets w.r.t. different video length in Figure 4. As shown in the figure, our model can achieve a compatible performance in terms of short videos but significant improvements on long videos.

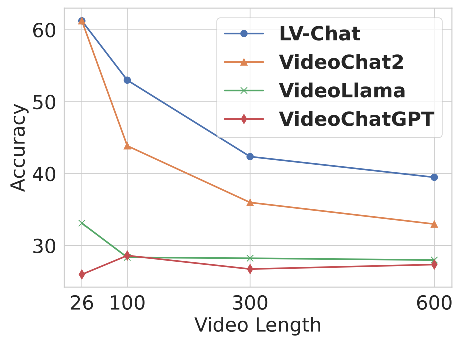


Figure 4: Average accuracies w.r.t different video lengths. "26" is the average duration of videos across four datasets. The IFE technique is not applied when videos are of lengths 26 and 100.

4.4 Ablation Study of LV-CHAT

We aim to study the effects of the finetuning of FSE (§ 3.2) and the IFE technique (§ 3.3). Thus we exclude these two parts in LV-CHAT to check the performance on the benchmarks. The results are reported in Table 2. From the table, we can observe that without IFE and FSE, the performance of LV-CHAT dropped, demonstrating the necessity of both

[2] https://github.com/DAMO-NLP-SG/Video-LLaMA

[3] https://github.com/mbzuai-oryx/Video-ChatGPT/blob/main/docs/offline_demo.md

	100s				300s				600s			
	AS	AP	UA	OI	AS	AP	UA	OI	AS	AP	UA	OI
VideoChatGPT	30	23	34	27.5	27.5	25.5	28	26	26	27	30	26.5
VideoLlama	24	23.5	39	27	25.5	23.5	38	26	23.5	25	37.5	26
VideoChat2	38.5	33	46.5	57.5	30.5	29	**45**	39.5	28.5	23	**41.5**	39
LV-CHAT	**53.5**	**45.5**	**47**	**66**	**42.5**	**37.5**	37	**52.5**	**37**	**34**	38.5	**48.5**

Table 1: Results on QA datasets extended from MVBench. The interleaving factor γ is set to be 2 for videos of length 5 min and 4 for videos of length 10 min. All models are evaluated using MVBench's protocol.

	100s				300s				600s			
	AS	AP	UA	OI	AS	AP	UA	OI	AS	AP	UA	OI
LV-CHAT	**53.5**	**45.5**	**47**	**66**	**42.5**	37.5	37	**52.5**	**37**	**34**	**38.5**	**48.5**
w/o IFE	-	-	-	-	41	**38.5**	**38.5**	47	34.5	30.5	38.5	46
w/o IFE, w/o FSE	35.5	33.5	36.5	43	32	28.5	28.5	39	27	28	28	35.5

Table 2: Ablation Study. We exclude IFE and FSE from LV-CHAT to study the effectiveness of these techniques.

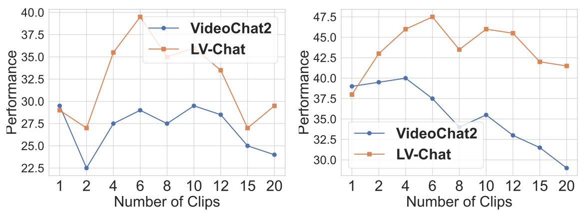
FSE and IFE in terms of long video understanding.

4.5 Model Analysis of LV-CHAT

As the performance of LV-CHAT across the Action Sequence (AS) and Object Interaction (OI) datasets are most pronounced, we focus on these two datasets to study the efficacy and properties of LV-CHAT.

4.5.1 LV-CHAT can handle more embeddings

Our investigation aims to evaluate the performance difference between LV-CHAT and our backbone about the number of clips, n. For this purpose, experiments were conducted with n ranging from 1 to 20, under a consistent video duration of 600 seconds. For instance, when $n = 4$, we sample $K * 4$ frames out of the entire video. The findings, illustrated in Figure 5, reveal two key insights: (1) LV-CHAT consistently surpasses the baseline performance across all tested clip counts, which demonstrates the robustness and enhanced capacity for longer video understanding; (2) We noticed that as we increase the number of clips, LV-Chat's performance gets better up to a certain point. Specifically, the model performs best with 6 clips. If we keep adding more clips beyond this number, up to 12, the performance barely drops. However, once we go over 12 clips, the performance begins to drop. This trend suggests that having too many clips doesn't always help as the model is trained with limited number of clips. This finding supports our earlier discussion about the challenges of matching the model's training experience with its usage in real-world scenarios, where the data it encounters can vary widely.



(a) Action Sequence (b) Object Interaction

Figure 5: Accuracies w.r.t. the number of tokens

4.5.2 Effectiveness of IFE

To assess how well Interleaved Frame Encoding (IFE) works, we tested it on videos of varying lengths: 100s, 200s, 300s, 400s, 500s, and 600s. For each video length, we adjusted the interleaving factor γ from 1 to 6, respectively. This setup aligns with our previous finding that LV-CHAT shows optimal performance with up to 6 clips (as detailed in § 4.5.1). The results, summarized in Figure 6, indicate a clear trend: incorporating IFE improves the model's performance. Notably, as the video length increases, the benefit of using IFE becomes even more pronounced. Detailed performance metrics across four datasets are provided in § C.2.

4.5.3 Varying K in FSE

As shown in our main experiments (§ 4.3), the number of frames per clip is set as $K = 16$. We aim to show converting 16 frames into $N = 96$

	100s				300s				600s			
	AS	AP	UA	OI	AS	AP	UA	OI	AS	AP	UA	OI
LV-CHAT ($K = 8$)	48.5	44.0	42.5	61.0	**43.5**	37	33.5	50	34	32	34.5	49
LV-CHAT w/o IFE ($K = 8$)	-	-	-	-	42.5	35.5	36	49.5	34	32	34.5	49
LV-CHAT ($K = 16$)	**53.5**	**45.5**	**47**	**66**	42.5	37.5	37	**52.5**	**37**	**34**	**38.5**	**48.5**
LV-CHAT w/o IFE ($K = 16$)	-	-	-	-	41	**38.5**	**38.5**	47	34.5	30.5	38.5	46

Table 3: Ablation study with different K on long-video question answering benchmarks. **Bold**: best results.

	TACoS(287s)				EgoSchema(180s)
	Rouge1	Rouge2	RougeL	RougeSum	Accuracy
VideoLlama	0.269	0.0490	0.196	0.193	0.284
VideoChatGPT	0.263	0.0567	0.188	0.188	0.260
VideoChat2	0.261	0.0675	0.195	0.196	0.500
LV-CHAT	0.360	0.0920	**0.244**	**0.246**	0.554
LV-CHAT w/o IFE	**0.364**	**0.0931**	0.244	0.245	**0.560**

Table 4: Evaluation on long-video caption generation datasets. **Bold**: best results.

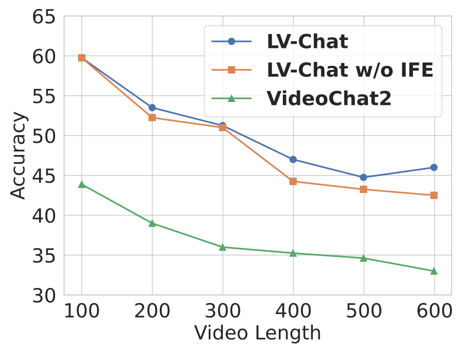


Figure 6: IFE effectiveness on QA datasets.

embeddings is not over-compression by checking the performance of LV-CHAT when we map every $K = 8$ frames into $N = 96$ embeddings. The results with $K = 8$ are reported in Table 3. From the table, we can see that $K = 16$ performs better than $K = 8$, showing that $K = 16$ may have not led to over-compression as $K = 8$ could not mitigate the potential over-compression problem. This observation is also partially observed in VideoChat2 (Li et al., 2023) where extracting 16 frames from the video and mapping them into 96 tokens generally perform better than extracting 8 frames.

4.6 Real World Datasets

In this section, we evaluate the performances of the baselines and LV-CHAT on two real-world datasets (by real-world, we mean the videos are naturally long videos, instead of extending short videos with unrelated ones):
TACoS (Rohrbach et al., 2014): This is a dataset comprising 127 videos averaging 287s and human-annotated captions of critical timestamps in the video. We use OpenAI's GPT-4 (OpenAI, 2024) to generate a reference summary from the labeled captions and conduct a human inspection (detailed in §B.4). Then all models are prompted to generate detailed descriptions and the ROUGE scores are calculated against the reference.
EgoSchema (Mangalam et al., 2023): Derived from Ego4D (Grauman et al., 2022), it is a long-form video question-answering dataset with an average length of 180; 500 samples with public released answers are used for evaluation.
For TACoS description generation, all models generate up to 100 new tokens using greedy search. For EgoSchema evaluation, we use the same settings as MVBench. The results are reported in Table 4. From the table, we can observe: (1) LV-CHAT outperforms all the other baselines by a large margin. Note that we have not done any fine-tuning on the datasets being tested (2) Although IFE does not provide any improvement over EgoSchema, we think one potential reason is that the lengths of the videos from this dataset are not sufficiently long (160 frames are sampled when IFE is not used, which is close to the average duration 180s.)

4.7 Case Study

We compare LV-CHAT against the baselines on the TACoS and show results in Table 5. For each video, we choose two representative scenes and match them with the captions from the TACoS.

Captioned scenes	Results
 He cut off ends of cucumbers.  He put cucumber on plate.	LV-CHAT: "The video shows a man standing in a kitchen and cutting a cucumber on a cutting board. He then places the cucumber on a plate and walks away. The kitchen has a counter with a sink and a refrigerator. The man is wearing a black shirt and is standing in front of a black and white fridge. The video is shot from an overhead angle, giving a bird's eye view of the kitchen..."
	VideoChat2: "The video shows a man cutting a cucumber on a cutting board."
	VideoChatGPT: "The video shows a person cutting vegetables on a countertop in a kitchen."
	VideoLlama: "1. A man is standing in a kitchen with a black countertop and a wooden table. 2. He is wearing a black shirt and a black jacket. 3. He is preparing food in the kitchen. 4. He is using a knife to cut a piece of meat. 5. He is cutting the meat on a cutting board. 6. He is using a spoon to stir the food. 7. He is standing in the kitchen"
 The man slices the broad beans.  The man continues to stir the beans as they cook.	LV-CHAT: "In the video, we see a man standing in a kitchen and preparing food. He is wearing a blue shirt and a white scarf around his neck. He is cutting vegetables on a cutting board and placing them in a pan. He then adds some oil to the pan and starts cooking the vegetables. The man is standing in front of a counter with a black and white marble top. There is a microwave oven on the counter, and a white bowl..."
	VideoChat2: "The video shows a man cutting vegetables on a cutting board."
	VideoChatGPT: "In the video, a woman is standing in a kitchen and preparing food. She is using a knife to chop vegetables on a cutting board, and she is also using a spoon to stir something in a bowl. The video also shows her using a spoon to stir something in a bowl."
	VideoLlama: "The video shows a man and a woman in a kitchen. The man is standing in front of a stove while the woman is standing in front of a counter. They are both wearing blue shirts. The kitchen has a black countertop and a black stove. There is a brown wooden table in the kitchen. The man is holding a knife and the woman is holding a plate. They are both preparing food."

Table 5: Two cases on the TACoS dataset of LV-CHAT compared with the baselines. The lengths of the two videos are 2 min 46 s and 11 min 11 s respectively. The highlighted parts are correct descriptions of actions.

We observe there are a number of cases where VideoChat2 can only summarize the whole video in one sentence without any further detail. And VideoChatGPT suffers from the same issue. While VideoLlama generates longer answers generally, it often has strong hallucinations on the details of the video and gives far-off descriptions. In contrast, our model captures much more details, including the actions of the subject and the environment where the video was shot. In the cases we show, we also highlight the correct action descriptions that these models generate. All three baselines fail to correctly capture the actions of the person from both two scenes while LV-CHAT succeeds in describing both. More comparisons between our model and VideoChat2 are shown in Appendix D.

5 Conclusion

In this study, we introduced Long Video Chat (LV-CHAT), a novel approach aimed at enhancing the comprehension capabilities of large language models (LLMs) for long video content. LV-CHAT has two innovative encoding strategies: Frame-Scalable Encoding (FSE) and Interleaved Frame Encoding (IFE). These techniques address the fundamental challenge of over-compression in video representation, a notable limitation in existing multimodal LLM frameworks when processing videos, particularly those exceeding one minute in duration. We evaluated LV-Chat's performance in long video comprehension tasks, utilizing both curated datasets and real-world benchmark. Our findings demonstrate that LV-Chat consistently surpasses previous methods in these settings.

6 Limitations

One limitation of LV-CHAT lies in the fine-tuning stage with the IFE enabled, which, contrary to expectations, did not yield any enhancements. This may be attributed to the insufficiency of long videos in the current video instruction dataset. Consequently, future work includes the development of datasets with longer videos to achieve better performances via fine-tuning. Another limitation is that LV-CHAT is based on VideoChat2 which uses Vicuna-7B-v1.0 as the LLM, which may be inferior than the most advanced LLMs such as Vicuna-7B-v1.5. Thus another future work is to train a larger model with more advanced LLMs, enhancing the understanding capabilities.

References

Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. 2021. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. 2023. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. 2022. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. In *CVPR*, pages 18009–18019. IEEE.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *CoRR*, abs/2306.15595.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Tsu-Jui Fu, Linjie Li, Zhe Gan, Kevin Lin, William Yang Wang, Lijuan Wang, and Zicheng Liu. 2021. Violet: End-to-end video-language transformers with masked visual-token modeling. *arXiv preprint arXiv:2111.12681*.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina González, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jáchym Kolár, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbeláez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard A. Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. 2022. Ego4d: Around the world in 3, 000 hours of egocentric video. In *CVPR*, pages 18973–18990. IEEE.

Michael Hassid, Tal Remez, Tu Anh Nguyen, Itai Gat, Alexis Conneau, Felix Kreuk, Jade Copet, Alexandre Defossez, Gabriel Synnaeve, Emmanuel Dupoux, et al. 2023. Textually pretrained speech language models. *arXiv preprint arXiv:2305.13009*.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. 2024. Llm maybe longlm: Self-extend llm context window without tuning. *arXiv preprint arXiv:2401.01325*.

Dongxu Li, Junnan Li, Hongdong Li, Juan Carlos Niebles, and Steven CH Hoi. 2022. Align and prompt: Video-and-language pre-training with entity prompts. In *CVPR*.

KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. 2023. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. 2024. Mvbench: A comprehensive multi-modal video understanding benchmark.

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

Ye Liu, Siyuan Li, Yang Wu, Chang Wen Chen, Ying Shan, and Xiaohu Qie. 2022. UMT: unified multi-modal transformers for joint video moment retrieval and highlight detection. In *CVPR*, pages 3032–3041. IEEE.

Ziyang Luo, Yadong Xi, Rongsheng Zhang, and Jing Ma. 2022. VC-GPT: visual conditioned GPT for end-to-end generative vision-and-language pre-training. *CoRR*, abs/2201.12723.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. 2023. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*.

Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. 2023. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Antoine Miech, Dimitri Zhukov, Jean Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *ICCV*.

OpenAI. 2024. ChatGPT (4) [Large language model]. `https://chat.openai.com`.

Ofir Press, Noah A Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Bharathkumar Ramachandra and Michael Jones. 2020. Street scene: A new dataset and evaluation protocol for video anomaly detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2569–2578.

Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. 2014. *Coherent Multi-sentence Video Description with Variable Level of Detail*, page 184–195. Springer International Publishing.

Amitay Sicherman and Yossi Adi. 2023. Analysing discrete self supervised speech representation for spoken language modeling. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2023. A length-extrapolatable transformer. In *ACL (1)*, pages 14590–14604. Association for Computational Linguistics.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR.

Alex Jinpeng Wang, Yixiao Ge, Rui Yan, Yuying Ge, Xudong Lin, Guanyu Cai, Jianping Wu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. 2022. All in one: Exploring unified video-language pre-training. *arXiv preprint arXiv:2203.07303*.

Junke Wang, Dongdong Chen, Chong Luo, Xiyang Dai, Lu Yuan, Zuxuan Wu, and Yu-Gang Jiang. 2023a. Chatvideo: A tracklet-centric multimodal and versatile video understanding system. *arXiv preprint arXiv:2304.14407*.

Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, et al. 2023b. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2307.06942*.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. 2021. Next-qa: Next phase of question-answering to explaining temporal actions. In *CVPR*, pages 9777–9786. Computer Vision Foundation / IEEE.

Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. 2022. Zero-shot video question answering via frozen bidirectional language models. In *NeurIPS*.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. 2020. CLEVRER: collision events for video representation and reasoning. In *ICLR*. OpenReview.net.

Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. 2021. Merlot: Multimodal neural script knowledge models. In *NeurIPS*.

Yan Zeng, Xinsong Zhang, Hang Li, Jiawei Wang, Jipeng Zhang, and Wangchunshu Zhou. 2022. X^2-vlm: All-in-one pre-trained model for vision-language tasks. *arXiv preprint arXiv:2211.12402*.

Hang Zhang, Xin Li, and Lidong Bing. 2023. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

A Notations

All the notations are provided in Table 6.

Symbols	Meanings
T	duration
F	total number of frames
K	number of frames in one clip
N	number of tokens per clip
F_s	number of sampled frames
n	number of clips
n_m	max number of clips
γ	number of interleaved times
n_i	number of clips in interleaved setting

Table 6: Notations

B Experiment Settings

B.1 Instruction Tuning Dataset Details

To fine-tune our model with FSE, we adopt the dataset collected by VideoChat2 (Li et al., 2023), where there is 1.9M video instruction data in total[4]. However, due to that some datasets are not accessible, we use a subset of this dataset:

- VideoChat (Li et al., 2023), collected from InternVid (Wang et al., 2023b).
- VideoChatGPT (Maaz et al., 2023), the original caption data is converted into conversation data by (Li et al., 2023).
- NExTQA (Xiao et al., 2021), a multi-choice question answering dataset.
- CLEVRER (Yi et al., 2020), an action prediction, multi-choice question answering dataset.

B.2 Datasets Selection Criteria

By manually looking at the examples, we compiled a few rules that a valid set of data should satisfy:

1. The baseline's performance drops as the target length of the extended video increases.
2. The baseline's performance should be better than random guesses.
3. Questions in the subset should not be greatly affected by video from Street-Scene.
4. Video should not be too short compared to our target length.
5. The questions in the subset should be answerable by a visual-only model. (i.e., the answers should not be all in the subtitles or the captions, leading to unanswerable questions based on visual data only)

By applying these rules, we select four datasets (Action Sequence, Action Prediction, Unexpected Action, Object Interaction) that are valid for testing long video-language models.

B.3 Dataset Extension

Despite the variety of videos that MVBench(Li et al., 2024) has. The average length of the four selected datasets are merely 25.5s, which can barely benefit from the capability of long-video models. To make use of these videos, we extend them with a second video sampled from the Street-Scene dataset(Ramachandra and Jones, 2020). The Street-Scene dataset contains 91 videos with 15 frames per second, and we select the first 54000 frames from the dataset, totaling an 1 hour video from which we sample the second video.

The extension process is as follows:

1. Set a target length of video T that the model should see.
2. For a original video v of length $\mathcal{L}(v) < T$, we applies a hash function $\mathcal{H}$ (see below) to the file name N_v of the video v to get a integer t_0 that is between 0 and 3600, which will be used as the starting time of the second video. The hash function in python is:

```
def hashstr(s: str) -> int:
  return sum(ord(c) * 31 ** (i % 3)
      for i, c in enumerate(s))
```

3. Draw a second video from the Street-Scene dataset that starts at $t_0 = \mathcal{H}(N_v)$ and ends at $t_0 + (T - \mathcal{L}(v))$.
4. Choose a time point $t_1 = \mathcal{H}(N_v + \text{":insert"})$ in the second video where we will insert the original video.
5. Insert the original video at t_1 of the second video and returns the extended video.

[4] https://github.com/OpenGVLab/Ask-Anything/blob/main/video_chat2/DATA.md

B.4 GPT-4 TACoS summarization

We use the following content to query the "GPT-4" API from OpenAI on Oct.9th, 2023. The context is composed of human-labelled captions and their starting times. The template we use for prompting GPT-4 is:

```
You are an assistant answering questions
     based on video contexts. Your
    answer should be based on the given
    contexts, but you can also infer the
     actual video content from the tag
    information and your common sense.
    The timed description is a
    description for the video at the
    given second. When describing,
    please mainly refer to the timed
    description. Don't create a video
    plot out of nothing.
Contexts for the video: \{context\}
Question: Could you please describe what
     is happening in the video?
```

Here is an example of video s13-d21. The prompt for GPT-4 is:

```
 You are an assistant answering
    questions based on video contexts.
    Your answer should be based on the
    given contexts, but you can also
    infer the actual video content from
    the tag information and your common
    sense. The timed description is a
    description for the video at the
    given second. When describing,
    please mainly refer to the timed
    description. Don't create a video
    plot out of nothing.
Contexts for the video: """
Second 9: He took out cutting board
Second 17: He took out knife
Second 22: He took out cucumber
Second 35: He took out plate
Second 47: He washed cucumber
Second 57: Cut off ends of cucumbers
Second 72: He sliced cucumbers
Second 90: He put cucumbers on plate
Second 9: person takes chopping board
    out
Second 17: person removes knife from
    draw
Second 22: person removes cucumber out
    of refrigerator
Second 35: person removes plate out of
    cabinet
Second 47: person then washes cucumber
Second 57: person then places cucumber
    on plate
Second 64: perosn then cuts ends off
    cucumber
Second 72: person then cuts cucumber in
    slices
Second 90: person then places cucumber
    on plate.
Second 9: The person gets out a cutting
    board.
Second 17: The person gets out a knife.
Second 22: The person gets out a
    cucumber.
Second 35: The person gets out a plate.
Second 47: The person rinses the
    cucumber.
Second 57: The person cuts the tips off
    the cucumber.
Second 96: The person slices the
    cucumber and puts the slices on the
    plate.
Second 9: The person gets out a cutting
    board.
Second 17: The person gets out a knife.
Second 25: The person gets out a
    cucumber.
Second 35: The person gets out a plate.
Second 47: The person rinses the
    cucumber.
Second 57: The person cuts off the tips
    of the cucumber.
Second 72: The person cuts up the
    cucumber.
Second 90: The person puts the cucumber
    slices on the plate.
Second 9: The person takes out a cutting
     board from the drawer.
Second 17: The person takes out a knife
    from the drawer.
Second 25: The person procures a
    cucumber from the fridge.
Second 35: The person procures a plate
    from the cabinet.
Second 47: The person washes the
    cucumber in the sink.
Second 57: The person cuts the ends off
    the cucumber then cuts the body into
     slices.
Second 90: The person sets cucumber
    slices on the plate.
Second 9: The person takes out a cutting
     board from the drawer.
Second 17: The person takes out a knife
    from the drawer.
Second 22: The person procures a
    cucumber from the fridge then takes
    a plate from the cabinet.
Second 47: The person washes the
    cucumber in the sink.
Second 57: The person cuts the ends from
     the cucumber.
Second 72: The person chops the cucumber
     into slices on the cutting board.
Second 90: The person sets the cucumber
    slices on the plate.
Second 9: The person takes out a cutting
     board from the drawer.
Second 17: The person takes out a knife
    from the drawer.
Second 22: The person procures a
    cucumber from the fridge.
Second 35: The person procures a plate
    from the cabinet.
Second 47: The person washes the
    cucumber in the sink.
Second 57: The person cuts the ends off
    the cucumber.
Second 72: The person slices the
    cucumber on the cutting board.
Second 90: The person sets the sliced
    cucumber on the plate.
Second 9: He goes to the drawer and
    takes out a cutting board and knife.
```

	AS	AP	AA	FA	UA	OE	OI	OS	MD	AL
VideoChat2	66	47.5	83.5	49.5	60	58	71.5	42.5	23	23
VideoChatGPT	23.5	26	62	22.5	26.5	54	28	40	23	20
VideoLlama	27.5	25.5	51	29	39	48	40.5	38	22.5	22.5
LV-Chat	62.5	47	79.5	44	61.5	56	74	40.5	23.5	27

	ST	AC	MC	MA	SC	FP	CO	EN	ER	CI
VideoChat2	88	39	42	58.5	44	49	36.5	35	40.5	65.5
VideoChatGPT	31	30.5	25.5	48.5	29	39.5	33	29.5	26	35.5
VideoLlama	43	34	22.5	45.5	32.5	32.5	40	30	21	37
LV-Chat	82	47.5	39.5	69.5	47	48.5	40	34.5	38.5	60

Table 7: Model Performance on the original MVBench. The results of VideoChat2, VideoChatGPT and VideoLlama are from the MVBench repository (https://github.com/OpenGVLab/Ask-Anything/blob/main/video_chat2/MVBENCH.md).

Second 25: He goes to the refrigerator and takes out a cucumber.
Second 35: He goes to the cupboard and takes out a plate and places it on the counter.
Second 50: He goes to the sink and washes the cucumber.
Second 57: He then cuts off the ends of the cucumber and then slices the cucumber.
Second 72: He picks up the cucumber and places it on the plate.
Second 9: He opens the drawers and takes out a cutting board and a knife.
Second 25: He gets a cucumber from the refrigerator and a plate from the cabinet.
Second 47: He sets the plate down and washes the cucumber in the sink.
Second 57: He puts the cucumber on the plate and dries off his hands.
Second 64: He uses the knife to cut off the ends of the cucumbers.
Second 72: He uses the knife to slice the cucumber into smaller pieces.
Second 96: He picks up the pieces of cucumber and places them on the plate.
Second 9: The person takes out a cutting board from the drawer.
Second 17: The person takes out a knife from the drawer.
Second 22: The person procures a cucumber from the fridge.
Second 35: The person procures a plate from the cabinet.
Second 47: The person washes the cucumber in the sink.
Second 57: The person chops the ends off the cucumber on the cutting board.
Second 72: The person slices the cucumber on the cutting board.
Second 90: The person sets the sliced cucumber on the plate.
Second 9: He gets out the cutting board, knife, plate, and cucumber from drawers and the refrigerator.
Second 50: He washes the cucumber in the sink and puts it on the plate.
Second 60: He wipes his hand on the rag.
Second 65: He chops the ends off the cucumber.
Second 72: He chops the cucumber into pieces with the knife.
Second 96: He gathers the chopped cucumbers together on a plate.
Second 9: the man is opening the drawer
Second 17: the man is placing a knife on the cutting board he got from the drawer
Second 22: the man is opening the refridgerator
Second 25: he got a cucumber from the refridgerator
Second 35: the man is washing the cucumber
Second 50: the man is slicing the cucumber width-wise
Second 96: then the man arranges the cucumbers on a plate
Second 9: The man places a cutting board on the counter.
Second 17: The man places a knife on the cutting board.
Second 22: The man gets a cucumber out of the refrigerator.
Second 35: The man gets a plate out of the cupboard.
Second 47: The man rinses the cucumber.
Second 57: The man dries his hands.
Second 64: The man cuts the ends off of the cucumber.
Second 72: The man cuts the cucumber into slices.
Second 96: The man arranges the cucumber slices on the plate.
Second 9: The person takes a cutting board from the drawer.
Second 17: The person removes a large knife from a drawer and puts it on the cutting board.
Second 25: The person takes a cucumber from the fridge.
Second 35: The person takes a plate from

```
    the cabinet.
Second 47: The person washes the
    cucumber in the sink.
Second 57: The person dries his hands
    with the towel.
Second 64: The person slices the
    cucumber.
Second 90: The person places the sliced
    cucumber onto the plate.
Second 9: He places the cutting board on
     the counter
Second 17: He placed the knife on the
    cutting board
Second 25: He got a cucumber from the
    refrigerator
Second 35: The man washed the cucumber
    in the sink
Second 57: The man dried his hands on
    the towel
Second 64: The man cut the cucumber into
     slices
Second 90: The man placed the cucumber
    slices on the plate
"""
Question:Could you please describe what
    is happening in the video?
```

And the response is:

```
The video is showing a person preparing
    a cucumber. They start by taking out
     a cutting board and a knife from a
    drawer. Then, they get a cucumber
    from the refrigerator and a plate
    from the cabinet. After that, they
    wash the cucumber in the sink. They
    then cut off the ends of the
    cucumber and slice it into pieces on
     the cutting board. Finally, they
    place the sliced cucumber on the
    plate.
```

We manually inspect the generated responses and query GPT-4 again if there are unreasonable responses or clear hallucinations.

B.5 Prompt for different models

B.5.1 System Prompt

For Video-ChatGPT, we use the system prompt (as used in the original paper):

```
You are Video-ChatGPT, a large vision-
    language assistant. You are able to
    understand the video content that
    the user provides, and assist the
    user with a variety of tasks using
    natural language. Follow the
    instructions carefully and explain
    your answers in detail based on the
    provided video.
```

For VideoChat2, Video-Llama, and our own model, we use the same system prompt from MVBench(Li et al., 2024):

```
Carefully watch the video and pay
    attention to the cause and sequence
    of events, the detail and movement
    of objects, and the action and pose
    of persons.
```

B.5.2 Dataset-specific prompt

In TaCoS generation, the user asks the assistant:

```
Based on your observations, describe
    what is happening in the video as
    detailed as possible.
```

In QA datasets (MVBench and EgoSchema), we use the same format as in MVBench. Following is an example:

```
Question: What happened after the person
     took the food?
Options:
(A) Ate the medicine.
(B) Tidied up the blanket.
(C) Put down the cup/glass/bottle.
(D) Took the box.
Only give the best option.
```

C Detailed Results

C.1 Model performance on all subsets of MVBench

Table 7 shows the results on the original MVBench and Table 8 shows the results on the augmented MVBench with Street-Scene.

C.2 Detailed Results for the Effectiveness of IFE

Table 9 shows the performance of LV-CHAT and VideoChat2 on the 4 chosen subsets extended to different lengths.

D Generation Cases

Some other TACoS generation cases are shown in Table 10. The Reference is the summary of the captions generated by OpenAI's GPT4.

Length 100s

	AS	AP	AA	FA	UA	OE	OI	OS	MD	AL
VideoChat2(16*1)	38.5	33	64.5	34	46.5	53	57.5	31.5	23.5	29
VideoChat2(16*10)	35.5	33.5	41.5	29.5	36.5	54.5	43	38	19.5	22
VideoChat2(8*10)	36.5	33	43	28	34.5	54	41.5	38	18.5	23
VideoChatGPT	30	23	54.5	24	34	53.5	27.5	41	24.5	26.5
VideoLlama	24	23.5	42.5	27	39	52.5	27	33	23.5	21
LV-Chat(8*10)	48.5	44	52.5	28.5	42.5	55	61	34	20.5	29
LV-Chat(16*10)	53.5	45.5	59.5	30	47	53	66	36.5	20.5	28

	ST	AC	MC	MA	SC	FP	CO	EN	ER	CI
VideoChat2(16*1)	72	43.5	30.5	57.5	54	29	40	31	39.5	43.5
VideoChat2(16*10)	40	39.5	22.5	37.5	58.5	26.5	38	24.5	30.5	39.5
VideoChat2(8*10)	40	38	22.5	37	57.5	27	41	25.5	32	44.5
VideoChatGPT	40	30	29	36.5	48.5	21	36	28.5	29	39
VideoLlama	32.5	29	28	41.5	45.5	29	34.5	30	25	35.5
LV-Chat(8*10)	55	39.5	26	46.5	48.5	31.5	39	37.5	35	39
LV-Chat(16*10)	62	41.5	27	49.5	47.5	28	36	38	37	38

Length 300s

	AS	AP	AA	FA	UA	OE	OI	OS	MD	AL
VideoChat2(16*1)	30.5	29	63	31.5	45	53	39.5	32	23	28.5
VideoChat2(16*10)	32	28.5	40.5	24	28.5	55.5	39	39	19	25
VideoChat2(8*10)	32	28.5	40.5	24	28.5	55.5	39	39	19	25.5
VideoChatGPT	27.5	25.5	54	23.5	28	53.5	26	43.5	24.5	29
VideoLlama	25.5	23.5	41.5	26.5	38	52	26	33	21.5	21
LV-Chat(8*10)	42.5	35.5	50	26.5	36	54	49.5	33.5	21.5	29
LV-Chat+IFE(8*10)	43.5	37	48.5	26.5	33.5	56	50	33	21	29.5
LV-Chat(16*10)	41	38.5	54	26.5	38.5	53.5	47	32.5	20.5	28.5
LV-Chat+IFE(16*10)	42.5	37.5	54	25	37	53.5	52.5	32.5	20	29

	ST	AC	MC	MA	SC	FP	CO	EN	ER	CI
VideoChat2(16*1)	60	44.5	28.5	58	57.5	27.5	41	33	35	42
VideoChat2(16*10)	36.5	38.5	22.5	37	58	25.5	38.5	25	26	39
VideoChat2(8*10)	36.5	38.5	22.5	37	58	25.5	38.5	25	26	39
VideoChatGPT	38.5	29.5	23.5	28	52	27	38	27	28.5	40.5
VideoLlama	30.5	29	28.5	41.5	47	29	33	32	22.5	34.5
LV-Chat(8*10)	51.5	39	25.5	45	48	29.5	34.5	36.5	30	34
LV-Chat+IFE(8*10)	46	40	28	46	48	29.5	35.5	36.5	29	33
LV-Chat(16*10)	49	37.5	29.5	45	48.5	27	34.5	36.5	35	34
LV-Chat+IFE(16*10)	48.5	39	29	47	48.5	29.5	30	35	32	35

Length 600s

	AS	AP	AA	FA	UA	OE	OI	OS	MD	AL
VideoChat2(16*1)	28.5	23	63	32	41.5	53	39	30.5	21.5	28.5
VideoChat2(16*10)	27	28	39	26.5	28	53	35.5	39	19	22.5
VideoChat2(8*10)	30	28	40	24.5	28.5	51	35.5	39	20.5	21.5
VideoChatGPT	26	27	56	25	30	52.5	26.5	40	24.5	25.5
VideoLlama	23.5	25	40	27	37.5	52.5	26	33	21.5	20
LV-Chat(8*10)	34	32	49	27.5	34.5	54	49	33	21.5	30
LV-Chat+IFE(8*10)	34	32	49	27.5	34.5	54	49	33	21.5	30
LV-Chat(16*10)	34.5	30.5	54	24	38.5	54	46	33.5	19	29.5
LV-Chat+IFE(16*10)	37	34	50.5	24.5	38.5	53.5	48.5	32.5	19.5	28.5

	ST	AC	MC	MA	SC	FP	CO	EN	ER	CI
VideoChat2(16*1)	51	45.5	28	59.5	56.5	30.5	36.5	33	32.5	43.5
VideoChat2(16*10)	38.5	38.5	22.5	36	57	26	39.5	25.5	25	38
VideoChat2(8*10)	35.5	38.5	23	33.5	59	26	37.5	24.5	25	36.5
VideoChatGPT	38	29.5	31	36.5	49	25.5	38.5	28.5	26.5	39
VideoLlama	28	29	29.5	42.5	47.5	29	33	31	22	33.5
LV-Chat(8*10)	42.5	42.5	26	43	48	30	33	36	29.5	35.5
LV-Chat+IFE(8*10)	42.5	42.5	26	43	48	30	33	36	29.5	35.5
LV-Chat(16*10)	44.5	37	24.5	46.5	48.5	27.5	35.5	36.5	33	35
LV-Chat+IFE(16*10)	47	41.5	24	47	47.5	27.5	37	36	35	33.5

Table 8: Model performance on extended MVBench

	AS	AP	UA	OI	Avg
Length 100s					
VideoChat2	38.5	33	46.5	57.5	43.875
LV-Chat(16*6)	54	42	48	65.5	52.375
LV-Chat(16*6+IFE)	51.5	44	58.5	64.5	54.625
Length 200s					
VideoChat2	35	29.5	44.5	47	39
LV-Chat(16*6)	44.5	42.5	48	60	48.75
LV-Chat(16*6+IFE)	47	43	47	60	49.25
Length 300s					
VideoChat2	30.5	29	45	39.5	36
LV-Chat(16*6)	46	34	41.5	56	44.375
LV-Chat(16*6+IFE)	46	39.5	42	56.5	46
Length 400s					
VideoChat2	30	26.5	44.5	40	35.25
LV-Chat(16*6)	38	36.5	40	50.5	41.25
LV-Chat(16*6+IFE)	40	38.5	40.5	54	43.25
Length 500s					
VideoChat2	28.5	27.5	46	36.5	34.625
LV-Chat(16*6)	38.5	38	41.5	48	41.5
LV-Chat(16*6+IFE)	40.5	36	39	49	41.125
Length 600s					
VideoChat2	28.5	23	41.5	39	33
LV-Chat(16*6)	37.5	35	43	47.5	40.75
LV-Chat(16*6+IFE)	43.5	34.5	41	48.5	41.875

Table 9: Accuracy on different video length

Video	Reference	LV-Chat	VideoChat2
s14-d46	The video is a tutorial on how to properly cut and prepare a pineapple. The woman starts by gathering all the necessary tools and the pineapple. She uses two different knives, a cutting board, a bowl, and a storage container. She begins by cutting off the top and bottom of the pineapple, then proceeds to cut off the sides using a smaller knife. She makes sure to remove all the hard spots on the pineapple's surface. After the pineapple is fully peeled, she slices it into rounds and places them in a stainless steel bowl. After she's done cutting, she cleans her knives and disposes of the pineapple scraps.	The video shows a woman standing in a kitchen cutting a pineapple on a cutting board. She is wearing a black shirt and a black apron. The kitchen has a black counter and a black sink. The woman is cutting the pineapple into slices with a knife. She is standing in front of a black stove and a black microwave. The video is shot from an overhead angle, giving a bird's eye view of the kitchen. The woman is cutting ...	The video shows a woman cutting a pineapple on a cutting board.
s36-d27	The video is a tutorial on how to properly cut and prepare an onion. The person in the video starts by taking out an onion, a cutting board, and a knife. They then cut off the ends of the onion and peel off the skin. After that, they cut the onion in half and peel off any remaining skin. The person then proceeds to dice the onion into small pieces. After the onion is diced, they get out a plate and put the diced onion on it. Finally, they add some lemon juice to the diced onion.	In the video, we see a man standing in a kitchen, looking at a plate of food. He then takes a knife and begins cutting the food on the plate. He continues to cut the food, and we see him cutting the food into smaller pieces. He then puts the food on a plate and puts it on the counter. He then takes a knife and begins cutting the food again. He continues to cut the food, and we see him cutting the food into smaller pieces...	The video shows a man cutting onions on a cutting board.
s36-d23	The video is a cooking tutorial where a man is preparing a carrot dish. He starts by washing his hands and gathering the necessary ingredients and tools, including carrots, a bowl, a cutting board, and a knife. He washes the carrots, then proceeds to chop them into small pieces. After chopping, he washes his cutting board and knife. He then turns on the stove, takes out a pan, and adds oil to it. He adds the chopped carrots to the pan and seasons them with various spices. He continues to cook the carrots, stirring them occasionally. Once the carrots are cooked, he transfers them to a bowl. Finally, he cleans up his workspace, washing and putting away the used utensils and equipment.	The video shows a man standing in a kitchen and preparing food. He is wearing a black shirt and a black and white checkered shirt. He is cutting vegetables and placing them in a pan. He then puts the pan on the stove and starts cooking. The video also shows a man standing in the kitchen and cutting vegetables. He is wearing a black shirt and a black and white checkered shirt. He is cutting vegetables and placing them ...	The video shows a man in a kitchen cutting vegetables and frying them in a pan.
s13-d21	The video is showing a person preparing a cucumber. They start by taking out a cutting board and a knife from a drawer. Then, they get a cucumber from the refrigerator and a plate from the cabinet. After that, they wash the cucumber in the sink. They then cut off the ends of the cucumber and slice it into pieces on the cutting board. Finally, they place the sliced cucumber on the plate.	The video shows a man standing in a kitchen, cutting vegetables on a cutting board. He then places the vegetables on a plate and puts it on a counter. The man then takes a knife and cuts the vegetables. He then puts the vegetables on a cutting board and cuts them again. He then puts the vegetables on a plate and puts it on the counter. The man then takes a knife and cuts the vegetables again. He then puts the vegetables ...	The video shows a man cutting vegetables on a cutting board.

Table 10: TACoS generation cases