HierarQ: Task-Aware Hierarchical Q-Former for Enhanced Video Understanding

Supplementary Material

In this supplementary material, we provide additional quantitative results in Section A. Then we present additional ablation studies in Section B, followed by qualitative analysis in Section C. Next, we provide more implementation details in Section D. Finally, we address some limitations and outline directions for future research in Section E.

A. Additional Results

In Table 9 we present the results for metadata classification task on the LVU dataset. The metadata classification task consists of - director, genre, writer, and year categories. Similar to the content understanding task presented in Table 1, HierarQ achieves state-of-the-art performance with an average performance gain of 5.5% over the best model [23] proving its the effectiveness across multiple categories of classification task. Additionally, in Table 10, we present performance comparison of our model against other concurrent approaches on the task of video captioning and report the METEOR scores.

Table 9. **Performance comparison of medium to long video understanding** on LVU dataset. The top-1 accuracy is reported. ‡ indicates without LLM finetuning. **Best** and second-best performances are highlighted.

Model	Director	Genre	Writer	Year	Avg
VideoBERT [62]	47.3	51.1	38.5	36.1	43.3
Obj_T4mer[77]	47.7	52.7	36.3	37.8	43.6
Orthoformer [47]	55.1	55.8	47.0	43.4	50.3
VIS4mer [28]	62.6	54.7	48.8	44.8	52.7
TranS4mer [29]	63.9	55.9	46.9	45.5	53.1
S5 [71]	67.3	65.4	51.3	48.0	58.0
Movies2Scene [12]	70.9	55.9	53.7	57.8	59.6
VideoMamba [37]	67.3	65.2	53.0	48.2	58.4
MA-LMM [23]	74.6	61.1	70.4	51.9	64.5
HierarQ ‡	76.6	66.2	71.1	59.2	68.3
HierarQ	**78.4**	**67.9**	**71.9**	**61.9**	**70.0**

B. Additional Analysis

Computational Cost Analysis. In Figure 8 we present detailed computation cost analysis of our method HierarQ and compare it against baseline Q-Former (BLIP-2 [34]) and few other concurrent approaches. Token: HierarQ uses a 32-token count, matching the token-count of MA-LMM and Video LLaMA while being smaller in tokens than baseline Q-Former (BLIP-2) and Video ChatGPT. Memory bank size: HierarQ uses fixed-size memory banks with 10 frames and 32 tokens per frame for both short- and long-term memory bank. In contrast, MA-LMM uses a variable-sized memory ranging from 10 to 40 frames with 32 tokens per frame, while MovieChat has 18 frames and 32 tokens per frame in short-term memory and 256 frames in long-term memory. Computation cost: HierarQ maintains constant memory consumption, processing over 10K frames on a 24GB A100 GPU (Figure 8 left). Unlike models with exponential computation growth, its auto-regressive design ensures scalability, fitting within a 24GB GPU for 100-frame inputs. Its training-free memory banks incur no additional computation costs, while the number of trainable parameters is only 390M. Latency: While latency increases linearly with frames (Figure 8 right), HierarQ remains capable of processing arbitrarily long videos, unlike other models that fail beyond a certain frame threshold.

Temporal modeling method ablation. Table 11 compares different temporal modeling methods. Some strategies to reduce token load due to auto-regressive frame processing is to do concatenation, average pooling or even token merging (ToMe) [6]. Our HierarQ outputs 32 tokens per frame, with simpler methods like concatenation and averaging of frame features yielding lower performance. Concatenation, in particular, is computationally expensive due to simultaneous processing of all frames. ToMe reduces tokens per frame from 32 to 2, but for 100-frame inputs, it still requires 200 tokens, imposing significant memory demands and sub-optimal performance. In contrast, our framework utilizes entity and scene-level task-aware streams with dedicated memory banks to store historical temporal information. The short-term memory bank employs a FIFO approach, while the long-term memory bank performs compression of similar token based on high cosine similarity (MBC). This temporal modeling technique keeps the token count fixed at 32 per frame, while not losing essential information and also optimizing GPU memory usage. Our temporal modeling approach achieves superior accuracy on the LVU and Breakfast datasets as compared to the other approaches. Here, we limit the number of frames to 100 for a fair comparison to the other methods that risk facing the LLM context length bottleneck with increased frames. Here, it is important to mention that previous works [23, 60] have successfully performed temporal modeling respectively using ToMe and MBC. However, here we empirically show that our combination of using short and long term memory for temporal modeling gives the best performance.

Long-term memory bank compression at different spa-

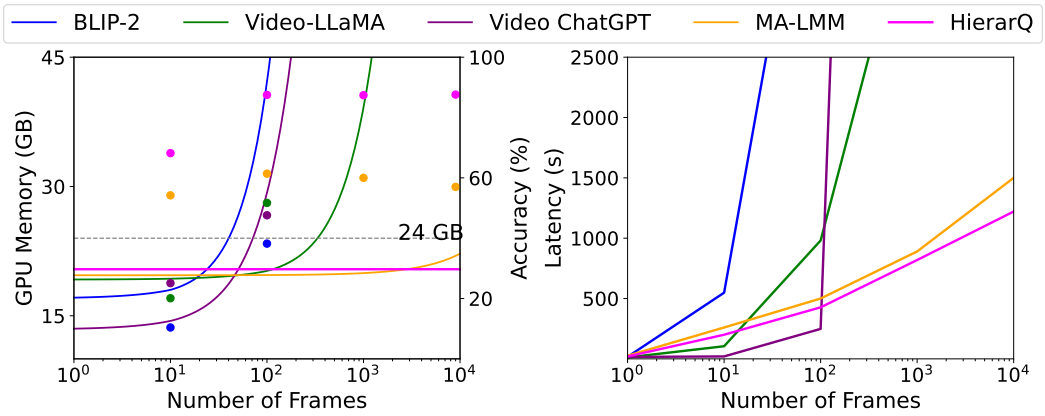

Figure 8. GPU memory (*line*), accuracy (*dots*) and latency vs. frames.

Table 10. **Performance comparison of video captioning.** Here we report the METEOR scores.

Model	MSRVTT	MSVD	YouCook2
SwinBERT [40]	29.9	41.3	15.6
GIT [70]	32.9	51.1	17.3
mPLUG-2 [80]	34.9	48.4	-
HowToCaption [59]	32.2	46.4	15.9
MA-LMM [23]	33.4	51.0	17.6
HierarQ	**35.1**	**51.2**	**18.1**

Table 11. **Ablation of different temporal modeling techniques.**

Method	#Frame	#Token	GPU	LVU	Breakfast
Concat	60	1920	53.1	65.1	94.2
Avg Pool	100	32	25.9	60.3	84.0
ToMe [6]	100	200	26.6	65.6	95.7
Ours	100	32	22.4	**67.9**	**97.4**

tial levels. Table 12 compares the performance of compressing the long-term memory bank at different spatial levels: frame-level and token-level, on the LVU and Breakfast datasets. In frame-level compression, cosine similarity is calculated between adjacent frame features, and features with the highest similarity are averaged. In token-level compression, cosine similarity is computed between tokens at the same spatial location across the entire temporal axis, leveraging the fact that each frame feature comprises multiple spatial tokens. We hypothesize that token-level compression preserves finer spatial details compared to frame-level compression. Proving our hypothesis, the results demonstrate that token-level compression consistently outperforms frame-level compression, supporting its ability to retain more detailed spatial information.

Feature modulation method ablation. To assess the task-aware feature modulation method's effectiveness, we conducted experiments using a randomly sampled subset of the LVU dataset's content-understanding videos, constituting 50% of the test set. We perform evaluation in two setups: frozen frames and out-of-distribution (OOD) frames. In the frozen frames setup, a single frame from each clip is repeated 10 times, testing the modulator's ability to identify task-relevant frames despite redundancy. In the OOD frames setup, 10 frames in each clip are replaced with frames from a randomly chosen sports video from YouTube [1], simulating irrelevant content to evaluate the modulator's filtering capabilities. Samples for both setups are shown in Figure 9.

Table 12. **Long-term memory bank compression strategy.**

Spatial Level	LVU	Breakfast
Frame-level	63.5	94.8
Token-level	**67.9**	**97.4**

We compare two modulation methods: CLIP scoring [52] and multi-headed attention. In the CLIP-based approach, frame relevance is scored and weighted against the text prompt, with higher scores indicating stronger relevance. As shown in Figure 10, the attention-based method outperforms CLIP scoring by better bridging the performance gap in both setups, demonstrating superior efficiency in filtering task-relevant frames and maintaining high performance across LVU dataset's all content understanding categories.

Effect of increasing the number of irrelevant frames. To further evaluate the impact of irrelevant frames on performance, we conducted experiments using the same randomly sampled subset of the LVU dataset's content-understanding videos as before. We maintained the original video length while introducing out-of-distribution (OOD) frames to extend the total duration of the video. We added OOD frames under four duration setups: 1.5 minutes, 3 minutes, 6 minutes and 9 minutes. These OOD frames were inserted in

[1] Youtube link of OOD sample.

Figure 9. **Sample frames from original video with irrelevant frames synthetically introduced to it** for evaluating the effectiveness of the task-aware feature modulator. Here, the *top* sample shows the frozen frames setup and the *bottom* sample shows the OOD frames setups. The red and blue boxes respectively denote the frozen and OOD frames.

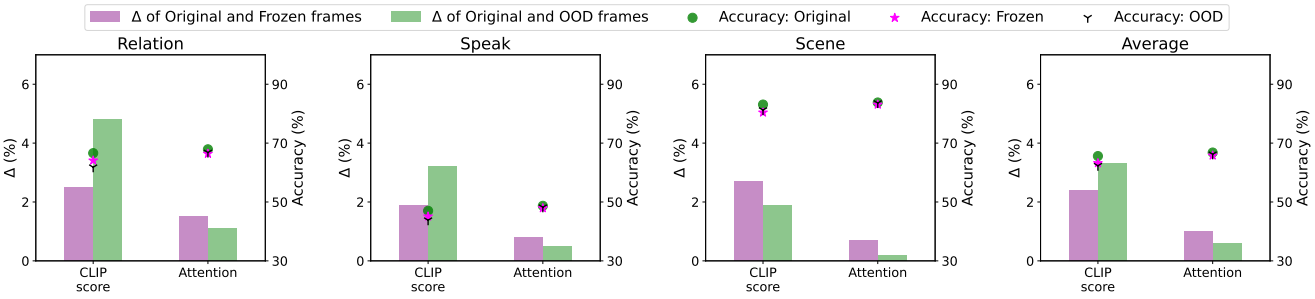

Figure 10. **Ablation of different feature modulation methods** on LVU dataset. Here, *left* y axis shows accuracy drop (Δ) between original and frozen/OOD frames and *right* y axis shows accuracy of original, frozen and OOD frames. Across all categories the attention mechanism bridges the performance gap more effectively than CLIP scoring while consistently getting high accuracy showing the effectiveness of the selected feature modulation method.

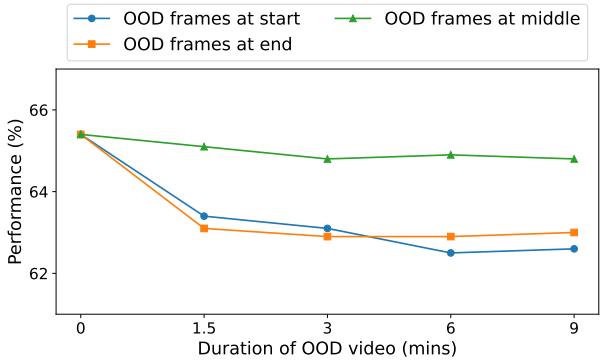

Figure 11. **Effect of increasing the number of irrelevant frames.** Here, y-axis denotes performance with irrelevant (OOD) frames inserted in different variations and x-axis denotes duration of OOD video where 0 denotes no OOD frames being inserted, e.g. original video.

three variations: at the beginning, at the end, and in the middle of the video.

As shown in Figure 11, the performance drop saturates even with an increasing number of OOD frames. This indicates that our feature modulator effectively filters out irrelevant information, thus helping HierarQ to maintain high performance despite the increased irrelevant video length. Furthermore, the position of OOD frames (beginning, middle, or end) shows neither significant nor conclusive impact on performance, demonstrating the robustness of our framework in modeling long-term temporal relationships irrespective of the position of irrelevant frames.

Effect of number of layers in the feature modulator. To assess the impact of varying the number of cross-attention layers in the task-aware two-stream feature modulators, we present the ablation results in Figure 12. The results indicate that increasing the number of layers initially improves performance, as more layers can model nuanced relationships effectively. Performance peaks at 2 layers for both LVU and Breakfast datasets, suggesting this is the optimal point for capturing task-relevant interactions without overfitting. Beyond 2 layers, performance plateaus and declines after 4 layers due to overfitting and diminishing returns, es-

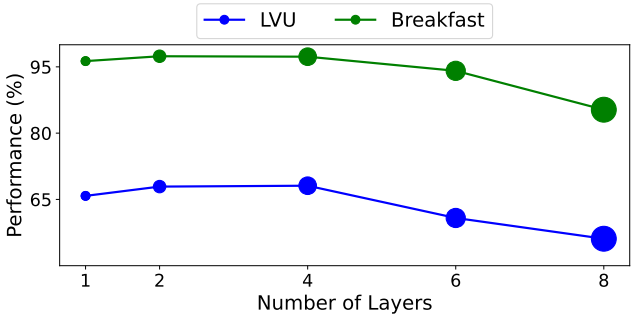

Figure 12. **Effect of number of layers in the feature modulators.** Here, the marker size denotes the number of parameters.

Table 13. **Performance comparison of different LLMs.** Here we report the top-1 accuracy for LVU and Breakfast and global accuracy for MovieChat-1k.

LLM	Size	LVU	Breakfast	MovieChat-1k
FlanT5-XL	3B	66.0	95.2	86.7
LLaMA-2	7B	67.3	97.1	87.2
Vicuna	7B	**67.9**	**97.4**	**87.5**

pecially as parameter count increases. The choice between 2 and 4 layers presents a trade-off: while 4 layers offer a marginal improvement on LVU, the linear increase in parameter count makes this configuration less efficient. Therefore, we adopt the 2-layer architecture as it strikes the best balance between performance and computational efficiency.

Choice of LLM. Our model supports various LLM architectures. To identify the best performer, we evaluated three popular LLMs: FlanT5-XL [15], LLaMA-2 [68], and Vicuna-7B [90]. As shown in Table 13, Vicuna-7B achieves slightly better performance than the others.

Robustness of entity guided feature modulator. Entities are extracted using POS tagging (*NN, NNS, NNP, NNPS*) with Python's NLTK library, covering all common and proper nouns. HierarQ uses long-term relationship modeling to focus on relevant entities. In tasks like video captioning, generic nouns (e.g., *"video"*) serve as entities, enabling the entity-guided feature modulator to process the entire video (Figure 13). If no entities are present, the modulator returns raw frame features to the short-term memory bank ensuring adaptability. For multiple non-distinguishing entities (Figure 12 in supplementary), the modulator processes all, while HierarQ prioritizes relevant relationships using the scene stream and long-term memory.

Motivation for a two stream architecture. The main motivation of the two-stream approach is to capture fine-grained entity details within a short context window separately from the broader scene-level understanding. The two streams complement each other (Table 6) by preventing key-information loss and effective extended temporal relationship modeling. Since actions are inherently tied to entities, the entity stream captures not only the entities but also their interactions within its focus window, and the scene stream situates those interactions in a global context. While a verb-focused stream is an interesting idea, testing its inclusion resulted in similar performance (0.53% drop on MSRVTT-QA) to the two-stream architecture.

For medium to long-context understanding, key information may be scattered across time and at risk of being lost due to memory constraints. While the scene-stream is based on the full prompt and captures global context, it risks losing crucial short-term entity-level details over extended time. The entity-stream mitigates this by providing entity-specific information within a shorter temporal window as a complementary signal (Table 6). Moreover, since it only focuses on entities in a short context, it is not over-crowded by other irrelevant information that might be present within that window. This balanced representation of local and global information enhances understanding.

C. Qualitative Analysis

In Figure 14, we present a qualitative comparison of HierarQ and MA-LMM on the long-video question answering task using the MovieChat-1k dataset, which is the longest dataset among our benchmarks. The results highlight HierarQ's superior task-aware video understanding capabilities over MA-LMM. The two-stream task-aware feature modulator enables effective entity- and scene-level understanding through its dual-stream design with dedicated memory banks which further supports HierarQ to effectively model the temporal relationship between short and long-term contexts.

For example, in the animal counting task, the entity-guided feature modulator along with the short-term memory bank helps track entity-specific details (e.g. "*animals*") across frames, while the prompt-guided feature modulator along with the long-term memory bank ensures continuity and accurate aggregation of historical information, enabling HierarQ to provide the correct answer. Similarly, the interplay between the entity- and scene-level Q-Formers inside the HierarQ allows nuanced reasoning, as seen in the "*man in the boat*" scenario, where HierarQ effectively models temporal relationships and historical context to deduce the correct outcome.

In contrast, MA-LMM lacks task-awareness and treats all frames equally, relying on coarse memory compression that leads to errors in tasks requiring detailed and contextual understanding. Even in tasks requiring whole-video analysis, such as identifying the presence of animals or stars, HierarQ excels by leveraging its task-aware design and superior historical information retaining capability with the help of two level of memory banks and hierarchical Q-formers. HierarQ demonstrates a superior ability to understand the

Figure 13. **Qualitative analysis of video captioning** on MSRVTT. Here, generic nouns (e.g., *"video"*) serve as entity and thus the entity guided feature modulator highlights the entire video.

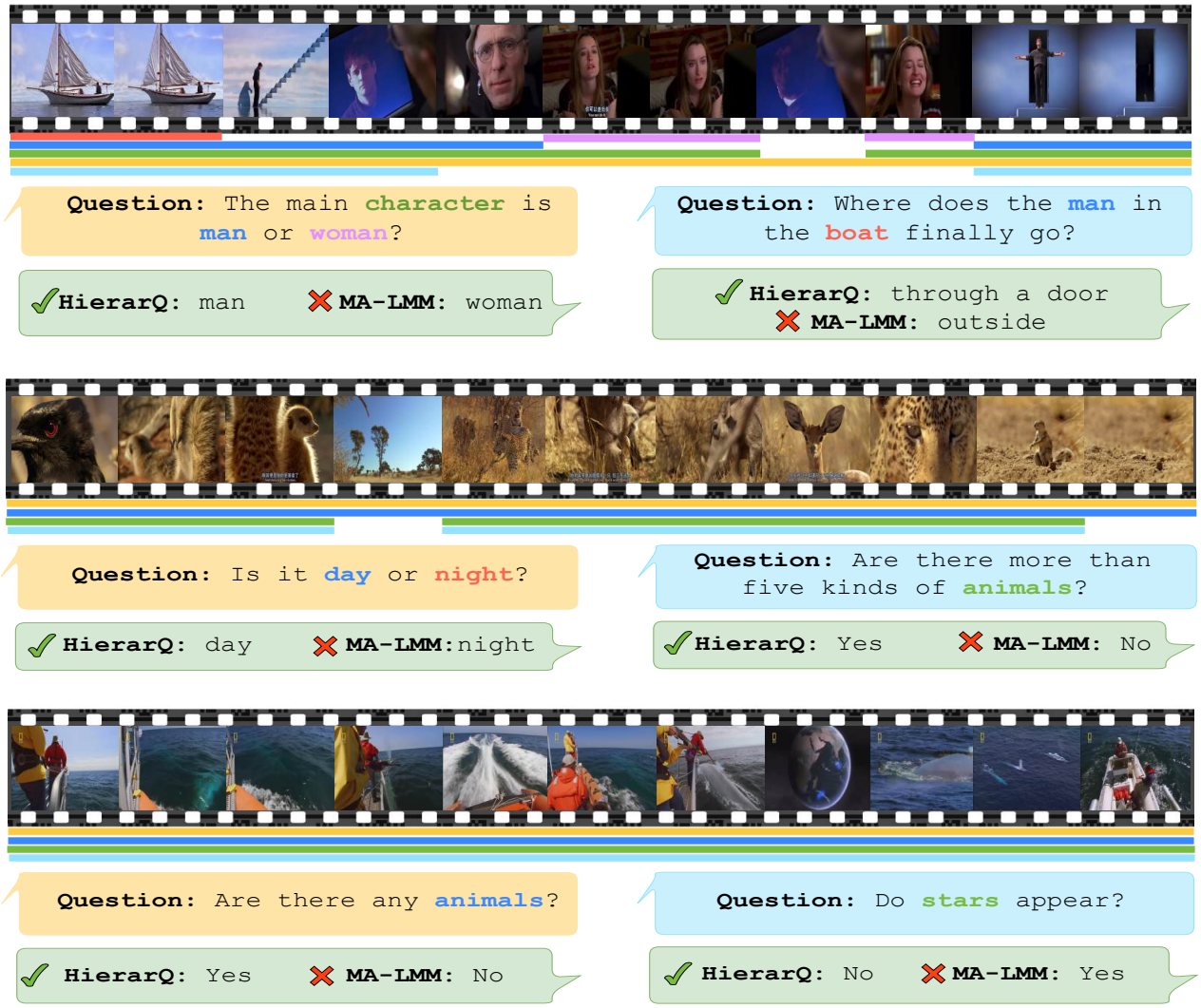

Figure 14. **Qualitative analysis of long-video question answering** on MovieChat-1k. Here, HierarQ adaptively focuses on task-relevant video segments, achieving a task-aware, comprehensive understanding. Color-coded frames are shown to demonstrate how entity-focused information complements the broader prompt-relevant context, enhancing overall video relevance and understanding.

video by analyzing it holistically, effectively identifying semi-rare events such as the appearance of the whale, which occurs only a few times. In contrast, MA-LMM might be missing out that information due to coarse compression across longer timeline without task awareness. Additionally, HierarQ accurately detects the absence of stars, avoid-

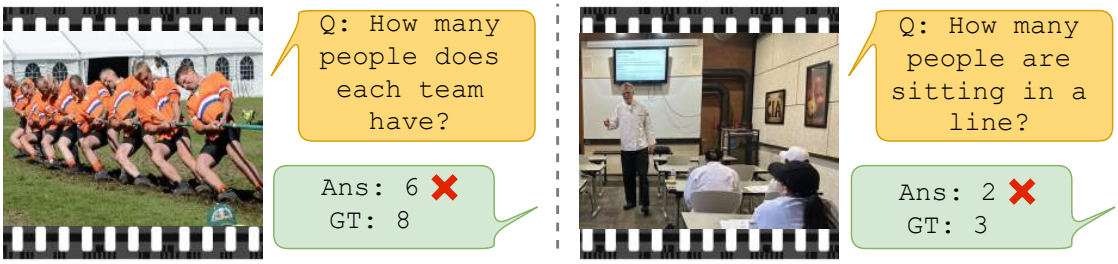

Figure 15. **Qualitative results of failure cases** on ActivityNet-QA.

ing the potential bias in MA-LMM that associates the presence of Earth with stars due to its focus on the planet's view from space. By leveraging task-aware modulation, a hierarchical Q-Former, and memory integration, our framework dynamically models both short- and long-term temporal relationships, enabling a more accurate and comprehensive understanding of videos.

On the contrary, Figure 15 highlights HierarQ's mispredictions due to ambiguous spatial arrangements.

D. Additional Implementation Details

Table 14 outlines the architectural details of our framework. The hidden size of the feature modulators is aligned with the ViT's hidden size to ensure compatibility. Similarly, the additional attention submodules in the Scene-level Q-Former of HierarQ maintain the same number of attention heads and hidden size as the previous layers for consistency.

Table 15 provides the hyperparameter details of the training setup. Across all experiments, we employ cosine learning rate decay, and the frozen ViT and LLM components are converted to FP16 precision to optimize performance. For evaluation, we adhere to standard protocols across datasets following [23]. We use 100 frames as input for all datasets. The train-test split for all dataset is presented in Table 16. One sample prompt for GPT-3.5-assisted evaluation is illustrated in Figure 16) which is used in long-video QA task evaluation.

Table 14. **Architectural details.**

Hyper-parameters	**Task-aware Feature Modulator**	
	Entity-guided	Prompt-guided
# of layers	2	2
# of attention heads	8	8
Hidden size	1408	1408
Hyper-parameters	**HierarQ**	
	Entity-level	Scene-level
# of layers	12	12
# of attention sub-modules	2	4
# of attention heads	12	12
Hidden size	768	768
Cross attention frequency	2	2
# of output query tokens	32	32
Memory bank length	10	10

Table 15. **Hyper-parameters for training.**

Hyper-parameters	Value
Patch size	14×14
Frame resolution	224×224
Training epoch	20
Batch size	32
Learning rate	$1e$-5
Weight decay	0.05
AdamW β	$[0.9, 0.999]$
LoRA rank	32
Beam size	5

E. Future Work

Future work could focus on developing dynamic memory management strategies that prioritize frames and scenes based on task relevance. By introducing adaptive memory update mechanisms, it would be possible to selectively compress or discard less relevant information, optimizing memory usage while maintaining performance. For example, task-aware memory filters could assess the importance of incoming features and dynamically decide whether to store or discard them, allowing the model to concentrate on the most critical temporal or spatial details. To further improve scalability and reduce inference time, processing videos in smaller chunks and modeling inter-chunk relationships through advanced techniques such as hierarchical attention or transformer-based methods could be explored. These enhancements would aim to improve the efficiency and effectiveness of video analysis tasks across various domains.

```
openai.ChatCompletion.create(
     model = "gpt-3.5-turbo",
     message = [
    {
      "role": "system",
      "content":
       "You are an intelligent chatbot designed for evaluating the correctness of generative
       outputs for question-answer pairs."
       "Your taks is to compare the predicted answer with the correct answer and determine
       if they match meaningfully. Here's how you can accomplish the task:"
       "-------"
       "## INSTRUCTIONS:"
       " - Focus on the meaningful match between the predicted answer and the correct
       answer. \n"
       " - Consider synonyms and paraphrases as valid matches. \n"
       " - Evaluate the correctness of the prediction compared to the answer."
    },
    {
      "role": "user",
      "content":
       "Please evaluate the following video-based question-answer pair: \n\n"
      f"Question: {question}\n"
      f"Correct Answer: {answer}\n"
      f"Predicted Answer: {pred}\n"
       "Provide your evaluation only as a yes/no and score where the score is an integer
       value between 0 and 5, with 5 indicating the highest meaningful match. "
       "Please generate the response in the form of a Python distionary string with keys
       'pred' and 'score', where value of 'pred' is a string of 'yes' or 'no' and value
       of 'score' is an INTEGER, not STRING."
       "DO NOT PROVIDE ANY OTHER OUTPUT TEXT OR EXPLANATION. Only provide the Python
       dictionary string."
       "For example, your response should look like this: {'pred': 'yes', 'score': 4.8}."
    }
  ]
)
```

Figure 16. **Prompt for GPT 3.5 assisted evaluation** for the long-video question answering task.

Table 16. **Dataset statistics.** Here QA pairs denote question-answer pair only applicable for video question answering task.

Dataset	Split	# Videos	# QA pair
Task: Video Understanding			
LVU	train	6927	-
	validation	1477	-
	test	1394	-
Breakfast	train	8451	-
	test	2816	-
COIN	train	9030	-
	test	2797	-
Task: Video Question Answering			
MSRVTT-QA	train	6513	158581
	validation	2990	12278
	test	497	72821
MSVD-QA	train	1200	30933
	validation	250	6415
	test	520	13157
ActivityNet-QA	train	3200	32000
	validation	1800	18000
	test	800	8000
MovieChat-1k	train	800	10400
	validation	100	1300
	test	100	1300
Task: Video Captioning			
MSRVTT	train	6513	-
	validation	2990	-
	test	497	-
MSVD	train	1200	-
	validation	250	-
	test	520	-
YouCook2	train	1333	-
	validation	457	-
	test	210	-