Temporal Chain of Thought: Long-Video Understanding by Thinking in Frames

Anurag Arnab* **Ahmet Iscen*** **Mathilde Caron** **Alireza Fathi** **Cordelia Schmid**
Google DeepMind

Abstract

Despite recent advances in Vision-Language Models (VLMs), long-video understanding remains a challenging problem. Although state-of-the-art long-context VLMs can process around 1000 input frames, they still struggle to effectively leverage this sequence length, and succumb to irrelevant distractors within the context window. We present Temporal Chain of Thought, an inference strategy for video question-answering that curates the model's input context. We use the VLM itself to iteratively identify and extract the most relevant frames from the video, which are then used for answering. We demonstrate how leveraging more computation at inference-time to select the most relevant context leads to improvements in accuracy, in agreement with recent work on inference-time scaling of LLMs. Moreover, we achieve state-of-the-art results on 4 diverse video question-answering datasets, showing consistent improvements with 3 different VLMs. In particular, our method shines on longer videos which would not otherwise fit within the model's context window: On longer videos of more than 1 hour on LVBench, our approach using a context window of 32K outperforms the same VLM using standard inference with a 700K context window by 2.8 points.

1 Introduction

Despite recent advances in Vision-Language Models (VLMs) [6, 31, 32, 34, 55], understanding long videos remains a challenging problem. This difficulty stems from the fact that this task requires a VLM to process a long sequence of input tokens, and requires the model to possess a host of interrelated abilities including action and scene understanding, long-term memory, and tracking state changes and interactions among others. The long-context ability of leading VLMs, that enables models to process hundreds or even a thousand frames of input context [36, 55, 56], is a valuable step forward in this regard. However, numerous studies have shown that processing longer contexts can saturate or degrade accuracy, as the model is overwhelmed with irrelevant or misleading content [23, 28, 37, 64].

Based on the observation that too large of an input context can be distracting, we propose an inference strategy, Temporal Chain of Thought, which first aggregates relevant context from the input video, and then uses it to answer the question (Fig. 1). Prior works based on the principle of removing distracting context from a video [28, 45, 61, 62] used an ensemble of multiple models, typically using one model to caption individual frames, another to find relevant ones, and finally answer the question with an LLM. In contrast, we use only a single VLM to both select the relevant context, and to answer the question, and show how our inference strategy provides substantial improvements.

Our approach is motivated by recent studies in Large Language Models (LLMs) which suggest that scaling inference-time computation is more effective than scaling the number of model parameters [9, 20, 52, 65]. Similarly, we show how leveraging more computation to aggregate relevant information from the video results in higher accuracy. In addition, as our approach iteratively extracts relevant

*Equal contribution.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

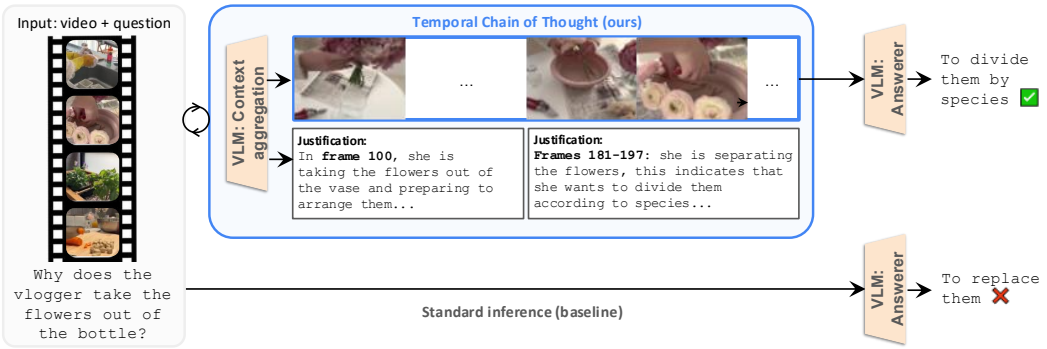


Figure 1: **Temporal Chain of Thought.** Motivated by the fact that long input contexts can have distractors which confuse the model, we use the VLM itself to first extract relevant context (blue box) before processing it. Our approach improves accuracy, and by iteratively processing parts of the video at a time, can also reduce the model's required context window.

context from the video, it means that we can effectively process videos that would not otherwise fit within the model's context limit. Moreover, our approach has connections to Chain-of-Thought prompting [63] (and multi-step extensions of it [59, 68]) in language, where the model is prompted to first output textual "thoughts" which help it to produce the final answer. As we aggregate the relevant frames in the video, we can think of these frames as "visual thoughts". Furthermore, as a by-product, we can use the model's justifications for choosing relevant frames to interpret it (Fig. 1).

We find that the general principle of aggregating relevant context from a video is beneficial for video question-answering (QA), with our proposed method consistently improving results across 4 datasets and 3 different VLMs. For shorter videos, on the order of hundreds of frames, our inference strategy improves results even when the entire video could fit within the model's context window, emphasising that by removing distractors from the input, we can improve the model's reasoning ability.

For longer videos, on the order of a thousand frames, such as LVBench [58], our method shows even larger improvements. Given a fixed context-window budget for a VLM, our model can iteratively extract the most relevant context leading to substantial accuracy gains. Furthermore, as our inference strategy focuses the model on the most relevant context, we can even outperform standard, baseline inference with a much longer context window.

In summary, we propose the following contributions:

- A novel VLM inference strategy for video QA.
- Thorough experimental analyses confirming that the principle of context aggregation is effective, that our approach outperforms standard long-context inference across a range of computational budgets, is adaptive to the question type and generalises to multiple VLMs.
- State-of-the-art results on 4 video understanding benchmarks. In particular, on LVBench where videos average 68 minutes in length, we improve by 11.4 points given a context-window budget of 32K tokens. Moreover, our iterative approach using a 32K context-limit outperforms a long-context baseline using the same 700K total tokens by 2.8 points.

2 Related Work

Long-context LLMs Our work is motivated by several prior studies, predominantly in the domain of natural language, which have demonstrated how Large Language Models (LLMs) are not able to effectively leverage their full input contexts [23, 37, 64, 67, 70, 74]. By asking questions where the position of a crucial piece of information in the input context is varied in a controlled manner, studies have shown that performance degrades considerably when the relevant context is not at the beginning or end of the input sequence [37, 64, 70].

Long-context video understanding The most related works for long-video understanding, however, are [5, 21, 28, 45, 57, 61, 62, 72]. These works represent a long video by first computing captions at each frame of the video with a dataset-specific model [22, 35, 73], which are then fed to an LLM along with the input question to produce an answer. Observing that redundant captions degrade the performance of the "answerer" LLM, a number of strategies have been proposed: Video Agent [61] begins from uniformly sampled frames, and uses EVA-CLIP embeddings [53] to iteratively retrieve frames until sufficient context has been obtained for GPT-4 to output an answer it is confident in. Video Tree [62] in contrast clusters per-frame captions together (where each cluster is then represented

by the caption of the frame closest to the centroid), and then only uses the top-scoring centroids in the final answering phase. Language Repository [28] aggregates per-frame captions into a global textual representation of the video, using an LLM to summarise different captions together, and CLIP similarities to prune redundant captions. VideoRAG [39], in contrast, supplements the video with auxilliary model outputs [24, 27, 46, 50], namely object detection, ASR and OCR, represented as text. Similarly, [16, 42, 54] call external tools based on per-frame captions to answer questions.

Although we also extract relevant context from the input video, our approach is fundamentally different in that we operate directly on video frames, and not captions as an intermediate representation. As we do not rely on initial per-frame captions, our approach is not limited by the captioner missing details relevant to the question (particularly because the captioning in these works is not conditioned on the input question). Moreover, we use only a single VLM in the entire inference process unlike the aforementioned works, which means that our approach is conceptually more elegant and simpler to deploy. The fact that we use a single model to generate intermediate outputs (the relevant frame indices to the question) is akin to "Chain-of-Thought" [29, 63] prompting.

Chain-of-Thought Chain-of-Thought (CoT) [63] was originally developed for few-shot prompting of LLMs for symbolic reasoning or arithmetic tasks. Concretely, the model is prompted to first output (in natural language) the steps first required to solve a reasoning problem, before outputting the final answer. This approach is effective as the LLM is conditioned on the initial reasoning (or "thoughts") before making its final prediction during autoregressive decoding. Subsequent works extended this approach, proposing inference-scaling strategies involving multiple LLM calls: The LLM first produces multiple hypotheses, which are then ranked, and the top-scoring ones explored further [9, 43, 51, 52, 59, 68]. Our method has similarities to [63], but instead of producing "thoughts" as language, predicts relevant frame indices in the video instead. Note that for video, Fei *et al*. [17] first predict spatio-temporal scene graphs [26] and use these intermediate representations for answering the question. Our method is more general, as it is not object-centric like [17], and our iterative approach for finding the most relevant frames has analogues of [52, 59, 68] for video understanding.

Token- or frame-selection Finally, numerous prior works learn neural network layers to reduce the number of tokens that need to be processed by a subsequent transformer [8, 14, 25, 48, 67, 75]. Among these, models specialised for video typically learn to select individual frames [10, 11, 30, 33, 49, 60]. However, these methods need to backpropagate gradients through the subsequent transformer during training, meaning that it is not computationally feasible to employ such approaches with the largest, pretrained VLMs [3, 55] as our work. Such models are also trained for a fixed, small number of input frames (*i.e*. 32 for SeViLA [69] and ViLA [60]), whilst we show that our approach can handle thousands. Moreover, perhaps surprisingly, we demonstrate that we do not need separate networks to select relevant frames, but can use the VLM itself to do so in an effective and adaptive manner.

3 Proposed Approach

We begin by reviewing the standard inference process for Vision Language Models (VLMs) before describing our proposed Temporal Chain of Thought.

3.1 Standard VLM inference

The standard inference approach for VLMs to answer a question, $\mathbf{q}$, about an input video, $\mathbf{x} \in \mathbb{R}^{T \times H \times W \times C}$, is to simply forward it through the model, f,

$$\mathbf{a} = f(\mathbf{x}, \mathbf{q}), \tag{1}$$

where f denotes the VLM, T, H and W denote the temporal- and spatial dimensions respectively, and $\mathbf{a}$ the predicted answer, where $\mathbf{q}$ and $\mathbf{a}$ are sequences of language tokens which index a discrete vocabulary, $\mathcal{V}$. Note that visual inputs, $\mathbf{x}$, are typically projected, or tokenised into the same space as language tokens. As the overall sequence length of the model is limited by computation, the frames of the video typically need to be subsampled to fit within the context-limit, k, of the model. Current models with the longest context windows can typically fit videos of up to one hour at 1 fps [36, 55].

3.2 Temporal Chain of Thought

Our method is motivated by the fact that although VLMs are now capable of handling increasing large input context lengths [36, 55, 56] they often still struggle to leverage this effectively and are confused by irrelevant distractors within this large context [23, 28, 37].

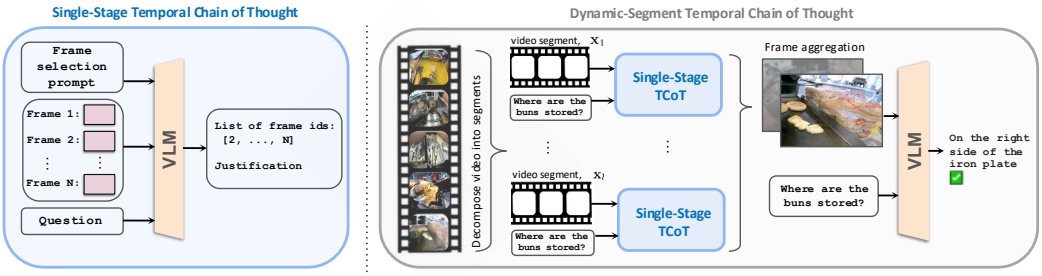


Figure 2: **Temporal Chain of Thought**. We use Single-Step TCoT (left, Sec. 3.2) to construct our final approach (right). Namely, we use the VLM itself to extract relevant frames from an input video clip, conditioned on the input question. To scalably process longer videos, we perform this approach within l segments which span the video to extract the most relevant context. Finally, we use only the extracted context for answering.

Therefore to answer a question, $\mathbf{q}$, about an input video, $\mathbf{x}$, we do not directly pass both inputs to the model directly. Instead, we decompose video question-answering into first extracting the relevant context, $\mathbf{c}$, from the input $\mathbf{x}$, and then answering using $\mathbf{c}$ instead (Fig. 1). Crucially, this decomposition is performed by the same instruction-tuned VLM which will perform the subsequent answering, drawing inspiration from Chain-of-Thought and related LLM inference strategies [52, 63, 68].

Formally, our inference procedure consists of two stages: First we assemble the relevant context, $\mathbf{c}$, from the input video $\mathbf{x}$ and question $\mathbf{q}$. Thereafter, we answer the question. We denote these stages as

$$\mathbf{c} = G(\mathbf{x}, \mathbf{q}), \tag{2}$$

$$\mathbf{a} = H(\mathbf{c}, \mathbf{q}), \tag{3}$$

where we denote G as the context aggregation function, and $H = f(\mathbf{c}, \mathbf{q})$ is the answering function, which simply forwards the extracted context through the VLM.

Note that G itself can be a multi-step inference process. It is also adaptive in that the number of context tokens, $\mathbf{c}$, that are selected is input-dependent as long as it fits within the VLM's context limit, k. Importantly, we use the same VLM for both G and H and do not rely on any external models or tools. Note that by carefully designing G, we can also effectively process long videos which would otherwise not fit in the model's context limit.

We first introduce a simple form of G next, Single-Step Temporal Chain of Thought (TCoT), which we use to construct our final approach, Dynamic-Segment TCoT.

Single-Step TCoT In this simple approach (Fig. 2, left), which is the basis of our final method, we simply query the VLM for what frames it needs to answer a question. Given up to N frames from the input video, $\mathbf{x} = [x_1, \ldots, x_N]$, which fit the context-limit of the model, we prompt the model to output the frame ids which are relevant to answering the given question as

$$\hat{\mathbf{x}}, \mathcal{S}, \mathbf{j} = S(\mathbf{x}, \mathbf{q}), \tag{4}$$

where S denotes the VLM selection call (prompt in Fig. 3), $\mathcal{S}$ the selected frames, and $\mathbf{j}$ a textual justification of this decision, which can be used for interpreting the model's predictions (Fig. 1). $\hat{\mathbf{x}}$ is the resampling of the input $\mathbf{x}$ using the frame ids in $\mathcal{S}$, namely $\hat{\mathbf{x}} = \{x_i, \ldots, x_j\}$ for $i, \ldots, j \in \mathcal{S}$.

We validate $\mathcal{S}$ to ensure that the frame indices are in ascending order, contain no duplicates and are within bounds. For simple parsing, we prompt the model to predict in the JSON format [15], and for responses that fail to parse, we assume that $\mathcal{S} = [1, \ldots, N]$ is all frames within the video.

```
You will be given a question about a video and five
possible answer options.

FrameID 1:{frame1},. . . ,FrameID N:{frame N}
Question: {question}
Possible answer choices: {answer choices}

Return the frame ids which can answer the given
question.

Please use the following JSON format for your
output:
{'frame_ids': [List of integer frame IDs],
 'justification': Justification about your output}
```

Figure 3: **Prompt for our VLM selection call**, S, (Eq. 4).

Additionally, in practice, we found that the relevant frames, $\hat{\mathbf{x}}$ are sometimes too succinct, which may make it more difficult for the answerer, H, to answer from $\hat{\mathbf{x}}$ alone. For example, as shown in App. C, for the question, *On what floor is the washing machine?*, the VLM correctly localises the washing machine, but it is otherwise difficult to discern which floor of the house it is on. To remedy this issue, we also include a small number, u, of uniformly sampled frames from the original input, $\mathbf{x}$, denoted as $\mathbf{x}_{[u]}$, where $u \ll N$. Thus, the final context aggregated is $\mathbf{c} = \hat{\mathbf{x}} \cup \mathbf{x}_{[u]}$.

This simple approach removes redundancy in the input $\mathbf{x}$ which may otherwise confuse the model [10, 28, 37, 69]. However, it does not scale to long input sequences. This is because we need to simultaneously process all frames, which may exceed the model's context limits, and this subtask of finding relevant frames in a long video is itself prone to distractors. We address these issues next.

Dynamic-Segment TCoT To decouple the video length from the context limit of the model, and to overcome limitations in frame recall from uniformly sampling input frames to fit within the model's context limit, we design Dynamic-Segment TCoT. As shown in Fig. 2, we partition the videos into l separate segments, which we process independently, before aggregating them to form a holistic video representation.

To process the input video $\mathbf{x}$, comprising N frames, we divide it into l non-overlapping segments of equal length. We denote each segment as $\mathbf{x}_i = \{x_{(i-1)m+1}, \ldots, x_{im}\}$, where $i \in \{1, \ldots, l\}$ indexes the segment, and $m = N/l$ represents the number of frames per segment.

Note that we may have a large segment size, m when either the video is long (large N) or few segments (small l). This presents a challenge as firstly, the segment length may exceed the model's context limit, k. Second, larger segments may contain more distractors, making it harder to identify relevant frames. To mitigate these issues, we uniformly sample s frames from each segment $\mathbf{x}_i$, denoted as $\mathbf{x}_{i_{[s]}}$ This ensures a tractable number of frames are considered, balancing computational cost with temporal coverage. The sampled frames from each segment are then processed independently, and the resulting outputs are concatenated to form the final representation:

$$\hat{\mathbf{x}} = [S(\mathbf{x}_{1_{[s]}}, \mathbf{q}), \ldots, S(\mathbf{x}_{l_{[s]}}, \mathbf{q})]. \tag{5}$$

As we processed segments independently, it is possible that $\hat{\mathbf{x}}$ is larger than our context-window limit, k. Moreover, we also find it beneficial to include coarse, uniform context, $\mathbf{x}_{[u]}$ as before. Therefore, if $\hat{\mathbf{x}}$ contains more frames than k, we refine $\hat{\mathbf{x}}$ by uniformly sampling $m = k - u$ frames from it, and adding u uniform context frames. Therefore, the final context aggregated is $\mathbf{c} = \hat{\mathbf{x}}_{[m]} \cup \mathbf{x}_{[u]}$.

Discussion Partitioning a video into l segments ensures that we can process a long-video with a fixed computational cost, regardless of the video-length, N. For standard VLM inference, the cost grows with the length of the video, and is limited by the maximum supported context-limit, k. In contrast, the required context-length with our approach is always fixed at s, to process a total of $s \cdot l$ frames. Note that our method not only decomposes a video into smaller segments, but importantly reasons across them in the answering stage, as analysed in Tab. 4 of App. A.

Moreover, by varying the number of segments, l, we can smoothly increase both inference-computation and accuracy (which we show experimentally in the next section). These trends are in agreement with recent work in language [9, 52, 59] which also uses additional compute at inference time to solve challenging problems with LLMs.

4 Experiments

4.1 Experimental Setup

Models We use Gemini 1.5 Flash as our primary VLM, specifically the *Gemini-1.5-flash-002* checkpoint via the Vertex API [18]. This is because Gemini is already the state-of-the-art on a number of video question-answering (QA) datasets (Tab. 3), and it supports long context lengths for fair comparison of baseline inference to our method. Gemini uses 258 tokens per frame, and unless otherwise specified, we use a context budget of 32K tokens. This corresponds to 120 frames leaving sufficient remaining tokens for the input question. To show the generalisability of our method, we also use Qwen-2.5-VL [6] and GPT-4o-mini [1]. The prompts for all models are detailed in App. D. For all datasets, we sample videos at 1 frame per second (fps) following [55, 58, 61, 72].

Datasets We use the following long-video QA datasets:

Egoschema [41] is a popular benchmark derived from Ego4D [19]. It consists of 5-way multiple choice questions on videos which are 180 seconds long. We run ablations on the subset of 500 labelled examples, and also report results on the full set of 5000 videos via the evaluation server.

LVBench [58] is a recent dataset with an average length of 4080 seconds (68 minutes), and four multiple choice options. Videos are from YouTube, and since some are no longer available online, we include our full list of video ids in App. D. We use visual inputs only.

OpenEQA [40] is a recent dataset targeted at embodied QA for mobile agents. It adds open-ended questions to the HM3D [47] and ScanNet [13] datasets, where videos are on average 452 frames long. As the answers are open-ended, the standard protocol is to use an LLM, specifically GPT-4 [2], to score the predicted answer against the ground truth answer using a scale from 1 to 5. These results are then averaged and normalised to a score out of 100.

NExT-QA [66] is a popular video QA dataset with five multiple choice options. Videos are on average only 39.5 seconds long. However, we report results on this dataset to compare to prior works.

4.2 Ablation Studies

Context Aggregation Analysis Table 1 compares different context aggregation functions, G, (Sec. 3.2) on both Egoschema and LVBench. Baseline inference performs no context aggregation and answers the question directly. We use a 32K token context limit, corresponding to 120 frames.

Table 1: **Comparison of different context aggregation approaches**. All methods take 120 frames as input, using a 32K context-window for Gemini Flash. Our improvements are larger on the longer LVBench dataset.

	Egoschema	LVBench
Baseline inference	72.6	50.3
Single-step	74.8	48.3
Hierarchical	74.0	53.3
Dynamic-segment	**75.2**	**61.7**

In addition to Single-Step and Dynamic-Segment TCoT (Sec. 3.2), we consider an additional approach which can iteratively process a long video, which we denote Hierarchical TCoT. As detailed in App. D, we perform Single-Step TCoT iteratively: First we coarsely sample frames from the video, then once we have identified frames of interest, we sample nearby frames that were not initially considered and iterate our aggregation procedure until the selected context has not changed, or the maximum number of iterations have been reached.

On Egoschema, all context aggregation methods, including the single-step variant, improve over baseline inference. Egoschema videos are short (180 frames), and effectively fit into the model's context window. The fact that all approaches improve over the baseline emphasise the utility of curating the model's input context to remove distractors.

On LVBench, where videos are far longer (average of 68 minutes), our proposed Dynamic-Segment TCoT (with $l = 12$) shines. This approach is able to effectively consider the whole video whilst adhering to its 32K context limit per VLM call. Single-step TCoT does not improve over the baseline here; we posit this is due to processing only a sparse subset (120 of average of 4080) of the total frames, which is too few to identify the relevant frames. Hierarchical TCoT mitigates this problem by adopting a coarse-to-fine strategy, but it underperforms our partitioned aggregation as it may miss short events which are not present in the initial, coarse sampling of the video.

We used $s = 64$ frames, and $l = 12$ segments for our experiment, ablating this choice in App. A.

Computation vs accuracy analysis We analyse the trade-off between computational cost and accuracy of TCoT in Fig. 4 on LVBench. We consider two alternatives: First, we perform standard inference with Gemini using a larger context limit. And second, as an alternate inference-time scaling strategy, we use "self-consistency" [59], where we sample multiple predictions from the VLM and take the majority vote as the final answer. These multiple results are obtained by randomly sampling 120 frames from the input, and also by increasing the sampling temperature to 0.7 [59] to obtain diverse output. We use the total number of tokens processed by the VLM to measure computation, as it is directly proportional to the monetary cost of using the model via an API. Moreover, metrics such as GFLOPs and inference-time are not available when calling the model through an API.

Figure 4 shows that as we increase the number of segments, l, and therefore the total number of tokens / frames processed, the performance of TCoT increases smoothly, whereas standard baseline inference saturates at around 1000 frames (264K tokens). When processing a total of 2700 frames (700K tokens), we are able to achieve an improvement of 2.8 points (61.7 vs 58.9) at the same cost.

As our approach curates the relevant context from the input video, and LVBench videos are very long with an average of 4080 frames, TCoT is not as effective when the total number of frames considered is low. This is because there are not sufficient input frames to select the most relevant context from. Therefore, we observe the benefits of our approach over the long-context baseline after processing a total of 512 frames (132K tokens).

Self-consistency chain-of-thought prompting [59], achieves minimal improvements over just a single inference call, highlighting that inference strategies developed for language are not directly applicable to video, which requires specialised approaches as ours.

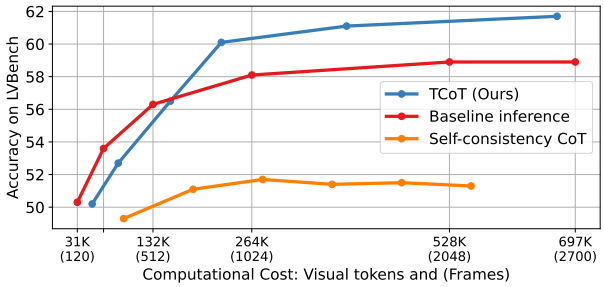


Figure 4: **Accuracy vs computation trade-off.** We compare Temporal Chain of Thought to two alternatives: baseline inference and self-consistency CoT [59]. We use the total number of visual tokens and frames (in parentheses) processed to measure computation, and vary l from 2 to 32 to do so for TCoT. Our approach improves consistently whilst baseline inference saturates in the presence of distractors from more frames. Self-consistency CoT is ineffective, underlining the need for inference-time scaling approaches tailored to video.

Note that as we use $s = 64$ frames, TCoT does not exceed 32K input context tokens, regardless of l and the total number of tokens / frames processed. Baseline inference, in contrast, is restricted by the context window supported by the model.

Alternate context aggregation approaches. Table 2 compares the following methods for choosing a fixed number of frames (120 to fit a 32K context limit) from a video:

- *Uniform sampling*: We uniformly select 120 frames.
- *Feature similarity*: We select frames based on their similarity to the embedded question using k-nearest neighbour search, as explored by [5, 16, 62]. We embed either the captions generated by our VLM ("question → captions") or the frames directly ("question → frames") using the strong dual-encoder SigLIP model [71]. For "question → captions", we prompt our VLM to generate concise captions to ensure they fit within the 64-token context limit of the SigLIP text encoder [71].
- *VLM-Based*: We use a VLM for selecting relevant context, either based on captions generated from the frames (as done by [21]), or by directly feeding the frames without any intermediate captions. We use the same partitioned segments (Sec. 3.2) in all cases.

Table 2: **Alternate context aggregation strategies**. We compare different methods for selecting 120 input frames. We use the same VLM, with 32K-context for the final answer.

	Selection strategy	Egoschema	LVBench
1	Uniform sampling	72.6	50.3
	Feature similarity		
2	Question → captions	73.8	52.1
3	Question → frames	73.4	54.4
	VLM-Based		
4	Select from "concise" captions	74.0	58.3
5	Select from "long" captions	72.8	60.4
6	Select directly from frames (Ours)	**75.2**	**61.7**
7	Oracle with annotated time references	–	67.4

We observe that all context aggregation approaches outperform naive uniform sampling on both datasets, highlighting the importance of curating the input to a VLM. Second, note that methods selecting frames directly with a VLM (rows 4–6) perform better than those relying on feature similarity (rows 2–3). Intuitively, this is because instruction-tuned VLMs can adapt easily to a new task in a zero-shot manner, and also dynamically vary the number of selected frames depending on the question type as shown in Fig. 6. Third, when selecting with a VLM, directly selecting from frames as in our method (row 6) performs better than using intermediate frame captions which lose information. Moreover, captioning-based approaches are sensitive to the prompt used for captioning: "concise" and "long" captions perform differently for Egoschema and LVBench (rows 4 and 5). Our captioning prompts are detailed in App. D. Finally, we observe that when we use the oracle time-reference frames which are human-annotated for LVBench [58], our performance increases by 5.7 points. The largest headroom, however, is in improving the answerer model (ie VLM) which is not in the scope of this work.

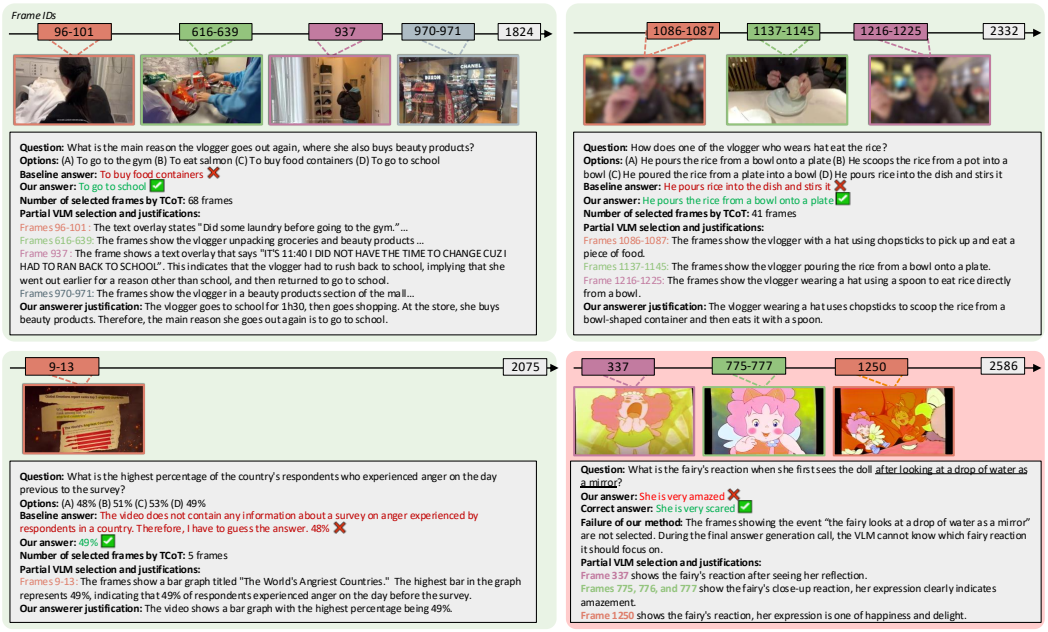


Figure 5: **Qualitative examples on LVBench.** Note how our model focuses on different parts of the video to make its prediction. For clarity, we sample frames from the segments selected by TCoT. In the failure case, although TCoT finds various frames showing the fairy's reactions, none of them include the drop of water, meaning that the answerer cannot make the correct prediction.

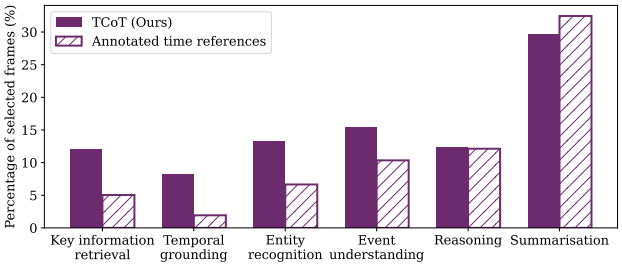


Figure 6: **Percentage of selected frames by question type.** Observe how the proportion of selected frames dynamically adjusts to the question type, aligning well with the human-annotated time-reference frames on LVBench [58].

Dynamic Context Aggregation Figure 6 analyses the percentage of selected frames across different question types on LVBench. Observe how the proportion of selected frames adapts dynamically based on the question type. For example, temporal grounding questions focus on specific moments in time, and accordingly less than 10% of the frames are chosen. Summarisation leads to the highest proportion of selected frames, which intuitively agrees with the need for a broad understanding of the video. Note that our selected proportions are also in agreement with the proportion of human-annotated "time-reference" frames [58]. Appendix A shows that our selected frames show distinct behaviours across datasets, confirming that our approach is indeed adaptive. On Egoschema, we select a larger proportion of frames, which agrees with the fact that it was annotated by [41] to require looking at at least 56% of the video. Finally, App. A also analyses performance by question type, showing that TCoT shows the largest improvement over the baseline for "temporal grounding" and "key information retrieval" which requires precise localisation of relevant context.

Qualitative examples Figure 5 and App. C present both success and failure cases of our approach.

4.3 State-of-the-Art Comparison

Finally, we compare to the state-of-the-art on four video QA datasets in Tab. 3, focusing on long-video datasets with prior works leveraging VLMs.

LVBench We substantially outperform prior works on this recent, challenging dataset (Tab. 3) with an average length of 68 minutes. Note that our Gemini 1.5 Flash baseline that we used in our ablations (Sec. 4.2) was already state-of-the-art, and we achieved substantial improvements: Namely, we improve by 11.4 points with the same 32K context limit, and by 2.8 points when processing the same number of total tokens (700K) with our TCoT method.

To show the generalisability of TCoT to other VLMs, we also present results using Qwen 2.5 VL 7B [6] and GPT-4o-mini [1]. We run baseline inference for both models with the maximum number

Table 3: State-of-the-art comparison. We report our own Gemini 1.5 Flash baseline for Egoschema and LVBench as it outperforms [55, 58]. For LVBench and OpenEQA, we report the tokens used in a single context-window, and the total number of tokens processed. OpenEQA uses the "LLM-as-judge" protocol of using GPT-4 to evaluate the answer. †: Our reproduction.

Egoschema and Next-QA

Method	LLM / VLM	Egoschema Subset	Egoschema Full set	NeXT-QA Accuracy
LangRepo [28]	Mixtral	66.2	41.2	60.9
MoreVQA [42]	PaLM-2	–	51.7	69.2
Video Agent [61]	GPT-4	60.2	54.1	71.3
Video Agent [61]†	Gemini 1.5 Flash	65.6	–	–
LLoVi [72]	GPT-4	61.2	52.2	73.8
GPT-4V [3, 7]	GPT-4V	63.5	55.6	–
LVNet [45]	GPT-4o	68.2	61.1	72.9
MotionEpic [17]	Vicuna	–	–	76.0
VideoTree [62]	GPT-4	66.2	61.1	75.6
BOLT [38]	LLaVA-One	60.6	64.0	79.5
LongVU [49]	Qwen2-7B	–	67.6	–
Gemini Flash [55]	Gemini 1.5 Flash	72.9	67.8	80.0
TCoT (ours)	Gemini 1.5 Flash	**75.2**	**69.1**	**81.0**

LVBench

Model	Context	Total	Accuracy
VideoAgent [16]†	–	–	37.6
InternVL-2.5-78B [12]	–	–	43.6
Qwen-2.5-VL-7B [6]	128K	128K	46.1
GPT-4o-mini [1]	22K	22K	48.0
Gemini 1.5 Flash [55]	32K	32K	50.3
Gemini 1.5 Flash [55]	700K	700K	58.9
TCoT (Qwen-2.5-VL-7B)	128K	320K	49.1
TCoT (GPT-4o-mini)	22K	86K	53.5
TCoT (Gemini 1.5 Flash)	32K	672K	**61.7**

Open EQA

Model	Context	Total	LLM Match
Claude 3 [4, 40]	6K	6K	36.3
GPT-4V [3, 40]	4.2K	4.2K	55.3
Gemini 1.5 Flash	77.4K	77.4K	68.0
TCoT (Gemini 1.5 Flash)	32K	76.4K	**69.2**

of frames / tokens that it supports to obtain the strongest possible baseline. Namely, this is 1024 frames / 128K for Qwen 2.5 and 250 frames / 22K for GPT-4o-mini (the GPT API supports only 250 frames, even though it supports more text tokens in its context [1]). Our improvements with TCoT are consistent and considerable on these VLMs, especially since our iterative TCoT allows us to consider more frames than the model supports natively in its context window. In particular, our improvement over GPT-4o-mini is the largest, at 5.5 points, as it supports the smallest context window.

Egoschema Table 3 shows that we outperform prior works on Egoschema. We compare to prior works that used either VLMs or LLMs (in conjunction with per-frame captioners) [45, 61, 62, 72]. We also re-implement VideoAgent [61] (details in App. D) using the same Gemini Flash VLM. Methods which initially compute captions on each frame are bounded by the quality of these captions, which often miss details relevant to the question. The Gemini 1.5 Flash baseline that we used in our ablations (Sec. 4.2) was already state-of-the-art, and we improve upon it further with TCoT.

NExT-QA The videos in NExT-QA are the shortest from all of our evaluation datasets, and average only 39.5 seconds. Nevertheless, we use this dataset to compare to prior works on this dataset which have also leveraged VLMs or LLMs (operating on per-frame captions) in Tab. 3. There is less headroom for us to improve on this dataset, as the videos are short and the accuracy of our Gemini 1.5 Flash baseline is already high. Our results are still consistent with our other experiments and Egoschema. Namely, we outperform prior works and we reduce the relative error of our state-of-the-art Gemini 1.5 Flash baseline by 5.0%.

OpenEQA OpenEQA presents a different domain, as the questions-answer pairs were labelled for mobile, embodied agents. Moreover, the answers are open-ended and an LLM (GPT-4 [2]), is used to evaluate answers in the authors' protocol [40]. Consistent with other datasets, we outperform prior works and our strong Gemini 1.5 Flash baseline using 300 frames (77.4K tokens).

5 Conclusion

We presented Temporal Chain of Thought, an inference strategy for long-video-question answering in VLMs, motivated by the fact that VLMs are affected by redundant information in their input context [23, 28, 37]. Our approach curates the input context of a VLM by decomposing a video QA task into first adaptively finding the relevant frames in the video. We demonstrated the efficacy of this approach by achieving state-of-the-art results on 4 different datasets. In particular, our approach with an input context of 32K tokens outperformed a 700K token model by 2.8 points on the challenging LVBench datasets where videos are 68 minutes long on average.

Limitations Our approach, which we have shown to be effective across three different VLMs, nevertheless requires a model with good instruction-following capabilities in order to perform the selection function (Fig. 3) in a zero-shot manner. And whilst we have shown significant improvements from our inference strategy, future work could improve this further by training the model explicitly for this inference approach via reinforcement learning [20, 44].

References

[1] Open AI. Gpt-4o mini. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

[2] Open AI. Gpt-4 technical report. In *arXiv preprint arXiv:2303.08774*, 2023.

[3] Open AI. Gpt-4v(ision) system card. 2024.

[4] Anthropic. The claude 3 model family: Opus, sonnet, haiku. 2024.

[5] Kirolos Ataallah, Xiaoqian Shen, Eslam Abdelrahman, Essam Sleiman, Mingchen Zhuge, Jian Ding, Deyao Zhu, Jürgen Schmidhuber, and Mohamed Elhoseiny. Goldfish: Vision-language understanding of arbitrarily long videos. In *ECCV*, 2024.

[6] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. In *arXiv preprint arXiv:2502.13923*, 2025.

[7] Ivana Balažević, Yuge Shi, Pinelopi Papalampidi, Rahma Chaabouni, Skanda Koppula, and Olivier J Hénaff. Memory consolidation enables long-context video understanding. In *ICML*, 2024.

[8] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *ICLR*, 2023.

[9] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. In *arXiv preprint arXiv:2407.21787*, 2024.

[10] Shyamal Buch, Cristóbal Eyzaguirre, Adrien Gaidon, Jiajun Wu, Li Fei-Fei, and Juan Carlos Niebles. Revisiting the" video" in video-language understanding. In *CVPR*, 2022.

[11] Shyamal Buch, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. Flexible frame selection for efficient video reasoning. In *CVPR*, 2025.

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

[13] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017.

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

[15] Alexander Dunn, John Dagdelen, Nicholas Walker, Sanghoon Lee, Andrew S Rosen, Gerbrand Ceder, Kristin Persson, and Anubhav Jain. Structured information extraction from complex scientific text with fine-tuned large language models. In *arXiv preprint arXiv:2212.05238*, 2022.

[16] Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. Videoagent: A memory-augmented multimodal agent for video understanding. In *ECCV*, 2024.

[17] Hao Fei, Shengqiong Wu, Wei Ji, Hanwang Zhang, Meishan Zhang, Mong-Li Lee, and Wynne Hsu. Video-of-thought: Step-by-step video reasoning from perception to cognition. In *ICML*, 2024.

[18] Google. Vertex api. https://cloud.google.com/vertex-ai.

[19] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, 2022.

[20] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. In *arXiv preprint arXiv:2501.12948*, 2025.

[21] Songhao Han, Wei Huang, Hairong Shi, Le Zhuo, Xiu Su, Shifeng Zhang, Xu Zhou, Xiaojuan Qi, Yue Liao, and Si Liu. Videoespresso: A large-scale chain-of-thought dataset for fine-grained video reasoning via core frame selection. *arXiv preprint arXiv:2411.14794*, 2024.

[22] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *CVPR*, 2024.

[23] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? In *arXiv preprint arXiv:2404.06654*, 2024.

[24] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *TMLR*, 2022.

[25] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver IO: A general architecture for structured inputs & outputs. In *ICLR*, 2022.

[26] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as compositions of spatio-temporal scene graphs. In *CVPR*, 2020.

[27] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[28] Kumara Kahatapitiya, Kanchana Ranasinghe, Jongwoo Park, and Michael S Ryoo. Language repository for long video understanding. In *arXiv preprint arXiv:2403.14622*, 2024.

[29] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.

[30] Bruno Korbar, Yongqin Xian, Alessio Tonioni, Andrew Zisserman, and Federico Tombari. Text-conditioned resampler for long form video understanding. In *ECCV*, 2024.

[31] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. In *arXiv preprint arXiv:2407.07895*, 2024.

[32] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[33] Yicong Li, Xiang Wang, Junbin Xiao, Wei Ji, and Tat-Seng Chua. Invariant grounding for video question answering. In *CVPR*, 2022.

[34] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[35] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024.

[36] Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. In *ICLR*, 2024.

[37] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[38] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding. In *arXiv preprint arXiv:2503.21483*, 2025.

[39] Yongdong Luo, Xiawu Zheng, Xiao Yang, Guilin Li, Haojia Lin, Jinfa Huang, Jiayi Ji, Fei Chao, Jiebo Luo, and Rongrong Ji. Video-rag: Visually-aligned retrieval-augmented long video comprehension. In *arXiv preprint arXiv:2411.13093*, 2024.

[40] Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, et al. Openeqa: Embodied question answering in the era of foundation models. In *CVPR*, 2024.

[41] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *NeurIPS*, 2023.

[42] Juhong Min, Shyamal Buch, Arsha Nagrani, Minsu Cho, and Cordelia Schmid. Morevqa: Exploring modular reasoning models for video question answering. In *CVPR*, 2024.

[43] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. In *arXiv preprint arXiv:2402.07872*, 2024.

[44] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[45] Jongwoo Park, Kanchana Ranasinghe, Kumara Kahatapitiya, Wonjeong Ryoo, Donghyun Kim, and Michael S Ryoo. Too many frames, not all useful: Efficient strategies for long-form video qa. In *arXiv preprint arXiv:2406.09396*, 2024.

[46] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *ICML*, 2023.

[47] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In *arXiv preprint arXiv:2109.08238*, 2021.

[48] Michael S Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? In *NeurIPS*, 2021.

[49] Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, et al. Longvu: Spatiotemporal adaptive compression for long video-language understanding. In *arXiv preprint arXiv:2410.17434*, 2024.

[50] Yunhang Shen, Chaoyou Fu, Peixian Chen, Mengdan Zhang, Ke Li, Xing Sun, Yunsheng Wu, Shaohui Lin, and Rongrong Ji. Aligning and prompting everything all at once for universal visual perception. In *CVPR*, 2024.

[51] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*, 2023.

[52] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. In *arXiv preprint arXiv:2408.03314*, 2024.

[53] Quan Sun, Jinsheng Wang, Qiying Yu, Yufeng Cui, Fan Zhang, Xiaosong Zhang, and Xinlong Wang. Eva-clip-18b: Scaling clip to 18 billion parameters. In *arXiv preprint arXiv:2402.04252*, 2024.

[54] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *ICCV*, 2023.

[55] Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. In *arXiv preprint arXiv:2403.05530*, 2024.

[56] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. In *arXiv preprint arXiv:2409.12191*, 2024.

[57] Shijie Wang, Qi Zhao, Minh Quan Do, Nakul Agarwal, Kwonjoon Lee, and Chen Sun. Vamos: Versatile action models for video understanding. In *ECCV*, 2024.

[58] Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Shiyu Huang, Bin Xu, Yuxiao Dong, Ming Ding, et al. Lvbench: An extreme long video understanding benchmark. In *arXiv preprint arXiv:2406.08035*, 2024.

[59] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[60] Xijun Wang, Junbang Liang, Chun-Kai Wang, Kenan Deng, Yu (Michael) Lou, Ming Lin, and Shan Yang. Vila: Efficient video-language alignment for video question answering. In *ECCV*, 2024.

[61] Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. Videoagent: Long-form video understanding with large language model as agent. In *arXiv preprint arXiv:2403.10517*, 2024.

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

[63] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[64] Tsung-Han Wu, Giscard Biamby, Jerome Quenum, Ritwik Gupta, Joseph E Gonzalez, Trevor Darrell, and David M Chan. Visual haystacks: Answering harder questions about sets of images. In *arXiv preprint arXiv:2407.13766*, 2024.

[65] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. In *arXiv preprint arXiv:2408.00724*, 2024.

[66] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *CVPR*, 2021.

[67] Jiaqi Xu, Cuiling Lan, Wenxuan Xie, Xuejin Chen, and Yan Lu. Retrieval-based video language model for efficient long video question answering. In *arXiv preprint arXiv:2312.04931*, 2023.

[68] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023.

[69] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. In *NeurIPS*, 2024.

[70] Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, et al. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. In *arXiv preprint arXiv:2402.05136*, 2024.

[71] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

[72] Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple llm framework for long-range video question-answering. In *EMNLP*, 2024.

[73] Yue Zhao, Ishan Misra, Philipp Krähenbühl, and Rohit Girdhar. Learning video representations from large language models. In *CVPR*, 2023.

[74] Zijia Zhao, Haoyu Lu, Yuqi Huo, Yifan Du, Tongtian Yue, Longteng Guo, Bingning Wang, Weipeng Chen, and Jing Liu. Needle in a video haystack: A scalable synthetic framework for benchmarking video mllms. In *arXiv preprint arXiv:2406.09367*, 2024.

[75] Xingyi Zhou, Anurag Arnab, Shyamal Buch, Shen Yan, Austin Myers, Xuehan Xiong, Arsha Nagrani, and Cordelia Schmid. Streaming dense video captioning. In *CVPR*, 2024.

Broader Impact

Our work presents an inference-strategy to improve the performance of Vision Language Models (VLMs) on long videos (our improvements are the largest on hour-long videos). VLMs, and video question-answering, the domain of our work, is a generic technology with a wide range of potential applications. We are unaware of all potential applications, but we are cognizant that each use-case has its own societal impacts depending on the intentions and motivations of the individuals or organisations building and deploying the system. For example, long-video understanding can be used for productive purposes, such as a user asking a system detailed questions about long, instructional videos. However, it may also be used for applications such as surveillance systems.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the paper are made at the end of the introduction section. Each of these claims are verified experimentally in the experimental section (Sec. 4). The novel VLM inference strategy that we proposed is contrasted to prior works in Sec. 2 and detailed in Sec. 3.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of our work are described in the Conclusion section (Sec. 5).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: There are no theoretical results in this work.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the main information required to reproduce experimental results in Sec. 4.1. Exhaustive experimental details are included in App. D. In particular, we have included all VLM prompts used in our experiments. We have used VLMs which can be publicly accessed via APIs or open-source weights, and have used publicly-available, academic datasets for our evaluation. For LVBench [58] which is a YouTube-based dataset, we have included a list of all Video IDs since videos can be removed from the platform.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We have used publicly-available academic datasets (Egoschema, Next-QA, LVBench and Open-EQA) for all of our experiments. To the best of our knowledge, the paper contains sufficient details to faithfully reproduce the experimental results. We are not, however, releasing our experimental code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental settings and details are described in Sec. 4.1 and App. D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Our experimental results in the ablation study (Sec. 4.2) are averaged over 3 runs, to control for random variability. And we therefore believe that our results and conclusions are signficant. We have not run further variations of our experiments as it is too computationally expensive to do so, as we are using VLMs for each experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: This is described in App. D. We call the Gemini 1.5 Flash [55] and GPT-4o-mini [1] through their public APIs. Qwen-2.5-VL [6] has publicly-available weights, and we run the HuggingFace implementation using a server with 8x NVIDIA A100 GPUs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: All authors have read the NeurIPS Code of Ethics, and attest that this research conforms in every aspect with it.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include a Broader Impact statement as the first section of the appendix, just before this checklist.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not release any data or models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have cited the original authors or all models and datasets used in the paper. We have complied with all available licenses for both models and datasets. We have used only publicly-available, academic datasets, for only academic purposes.

We include further information of each asset below:

- Next-QA [66] License: MIT
- LVBench [58]. License: CC BY-SA 4.0
- OpenEQA [40]. License: MIT
- Egoschema [41]. License: Unknown.
- Qwen-2.5-VL [6]. License: Apache 2.0

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not release any new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Our paper describes a novel inference strategy for VLMs for video data. Therefore, our paper describes our method, relation to prior work and experimental results with our approach in detail.

    However, we did not use an LLM at all to write the paper, or to write the code that we used to conduct experiments.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Table 4: **Independent segment answer aggregation**: As an additional baseline to show that our method can reason across different segments, we partition videos into segments, answer questions independently within each window, and aggregate these answers to predict the final response. This baseline substantially underperforms our final method (Sec. 3.2) as it cannot reason across different segments.

Model variant	LVBench
Individual segment answers	51.0
Individual segment answers with high confidence prompt	55.6
Ours	61.7

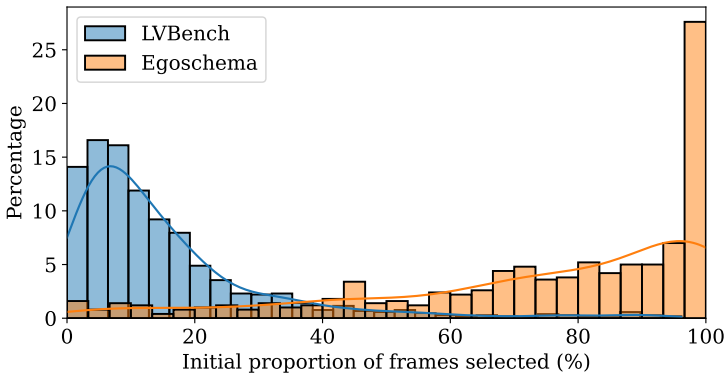


Figure 7: **Distribution of frames selected by our model**. Our model selects the relevant context in an adaptive manner, choosing a greater proportion of input frames for Egoschema (mean of 74%) than for LVBench (mean of 15%) The results on Egoschema correlate with its "temporal certificate" [41].

Appendix

In this appendix, we include additional experimental results (App. A), detailed failure mode analysis (App. B), qualitative examples (App. C), as well as additional experimental details (App. D). References follow the numbering from the original paper.

A Additional experimental results

Independent segment answer aggregation To show that our approach is able to reason across different segments in the video, we perform the following experiment with an alternate baseline: We divide videos from LVBench of N frames into $\lceil \frac{N}{s} \rceil$ segments of length s, and answer the question independently with a justification within each window. To compute the final answer, we pass each of the per-segment answers again to the VLM, and ask it predict the final answer. In Tab. 4, we observe that the performance is substantially lower on LVBench: 55.6% compared to our 61.7%. Intuitively, the final call to the VLM is often noisy because most segments are irrelevant. We found that the model should only propose an answer for a segment if it is highly confident ("high confidence prompt" in Tab. 4); otherwise, it attempts to answer for every segment, leading to contradictory and noisy information in the final VLM call. Additionally, this baseline underperforms compared to our method (Sec. 3.2) because many questions necessitate considering multiple segments. Therefore, our approach not only decomposes a video into smaller segments but also reasons across these segments, a capability lacking in simpler baselines.

Distribution of selected frames. To further analyse the adaptivity of our method in selecting frames relevant to the question, Fig. 7 analyses the proportion of frames selected. Concretely, we plot the ratio of frames initially selected by TCoT (Eq. 5). We note that if the number of selected frames is larger than the context-limit of the model, k, we reduce it by uniformly sampling within the sorted list of frame indices (Sec. 3.2). Nevertheless, it is more informative to analyse this initial distribution, as it shows the frames which the model deems relevant.

Our selection function shows distinct behaviour across Egoschema and LVBench, confirming that it is indeed adaptive to the input questions. The proportion of selected frames is significantly higher on

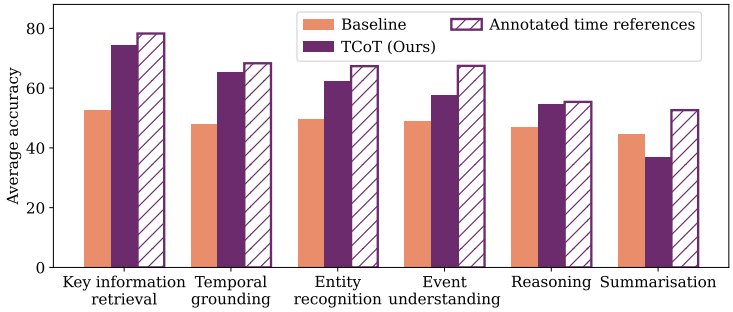


Figure 8: **Performance per question type on LVBench.** We compare baseline inference, our TCoT method, and using the oracle of the human-annotated time references to select relevant frames. We achieve significant improvements on most question types, and often near the accuracy of the oracle, particularly on "key information retrieval" and "temporal grounding" which requires precisely locating the relevant information in the long video.

Table 5: **Limits of Chain of Thought (CoT) methods in long video understanding**. Existing CoT techniques developed for language do not bring significant improvements on long video understanding. This calls for CoT techniques tailored to this task.

Chain of Thought (CoT) technique	# VLM calls	LVBench
None	1	49.5
Zero-shot CoT	1	50.3
Zero-shot CoT with 2-stage prompting [29]	2	49.4
Zero-shot CoT with self-consistency [59]	9	51.7

Egoschema, as its questions were designed by [41] to require a more holistic understanding of the video. Indeed the "temporal certificate" [41] (the duration of the video that an annotator must look at to answer the question) is estimated to be at least 100 seconds, or 56% of the video, which correlates with Fig. 7. Although LVBench videos are longer, their questions require localising specific moments in the video, as also shown in Fig. 7.

Performance per question type on LVBench. Figure 8 compares the performance of our method to baseline inference, and the oracle of using the annotated time-reference frames to select relevant frames. We observe that TCoT consistently outperforms baseline inference, particularly for "key information retrieval" and "temporal grounding" which requires precise localisation of relevant context. Moreover, we approach the accuracy of the oracle on these categories too. However, we note that baseline inference performs better for summarisation questions, which require a holistic view of the video. This is also the only question type in Fig. 6 where our method on average selects fewer frames than the oracle, suggesting that we do not select enough relevant information in these cases.

Limitations of pure text CoT on long video reasoning. Chain of Thought (CoT) prompting techniques enhance the reasoning capabilities of LLMs across various tasks. Inspired by these successes, we explore whether typical CoT methods can also improve performance in long video understanding in Tab. 5. In particular, we investigate:

- Zero-Shot CoT [29]: We add a sentence encouraging step-by-step thinking, namely "Explain your reasoning," into the prompt before outputting the final answer (Fig. 14).
- Zero-Shot CoT with two-stage prompting [29]: we append the output of the initial Zero-shot CoT to the prompt and generate a new prediction.
- Zero-Shot CoT with self-consistency [59]: we set the VLM temperature to 0.7 and sample a different set of frames at each run to encourage diverse reasoning across runs. The final prediction is determined by majority voting from nine runs, each employing Zero-Shot CoT. We experimented with more runs but observed no performance improvement (shown in Fig. 4).

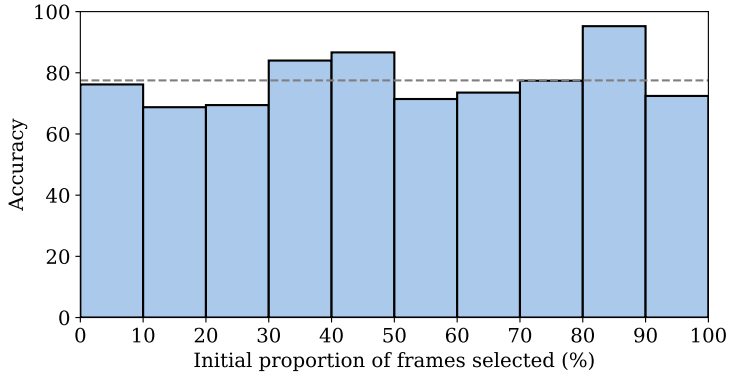


Figure 9: **Accuracy according to the proportion of frames selected on Egoschema**. Observe how our model's accuracy remains consistent, regardless of the number of frames selected, suggesting that our TCoT method can effectively and adaptively select the relevant frames to answer the question. The dashed line shows the overall average accuracy on the dataset.

Table 6: **Effect of hyperparameters**. We analyse the effect of the segment size, s (a), and the number of uniform context frames, u (b). The context-limit is $k = 120$, meaning that the remaining $m = k - u$ frames are selected by the model.

(a) Effect of segment size, s

s	Egoschema	LVBench
4	73.0	56.8
16	73.6	57.0
32	73.8	**58.1**
64	**75.2**	57.8
120	73.8	57.8

(b) Effect of uniform context, u.

m	u	Egoschema	LVBench
120	0	**75.2**	57.8
88	32	74.6	58.5
64	56	73.2	**59.3**
32	88	74.0	56.2
0	120	72.6	50.3

In Tab. 5, we observe that all the pure linguistic CoT techniques result in only marginal performance improvement on LVBench. This result calls for VLM inference strategies specifically tailored to video understanding tasks such as ours. Finally, note that we adopt "Zero-Shot CoT" as our default prompting technique for the VLM call generating the final answer (Fig. 14).

Accuracy is consistent across selected frames To further analyse the quality of our frame selection, Fig. 9 compares the accuracy of our TCoT model as a function of the proportion of frames initially selected.

If a model is effective in adaptively selecting only the frames that it needs to answer the question, then its accuracy should remain constant regardless of the number of frames selected. Figure 9 shows that this is largely the case for our TCoT model, highlighting that our approach effectively selects the frames it requires for the task.

Effect of hyperparameters, s and u Table 6 shows the effect of TCoT hyperparameters (Sec. 3.2), s (segment size) and u (uniform context).

Table 6a shows that the performance of our model is relatively consistent across different segment sizes, s, on both Egoschema and LVBench datasets. A smaller segment size means that we require a smaller context window during context aggregation, whilst also requiring more VLM calls. Note that the total context-limit to the answerer, remains constant at $k = 120$ here, enabling accurate predictions even with a small window, s.

Table 6b varies the amount of uniform context, u, that we add. Once again, the total context-limit remains fixed at $k = 120$, meaning that the remaining $m = k - u$ frames are selected by the model. Moderate amounts of uniform context help on LVBench, as they enable the model to obtain a broader awareness of the video. Egoschema, in contrast, does not benefit from uniform context, and this may be explained by Fig. 7 which shows that the model is already selecting a larger number of frames on this dataset, which has a "temporal certificate" [41] which covers the majority of the video. Finally, note that when $m = 0$, we are effectively performing the standard inference baseline as no frames are

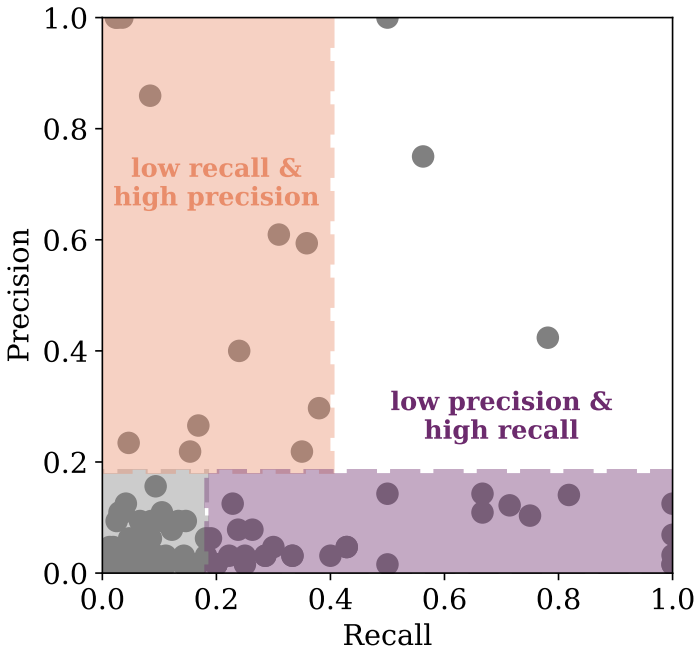


Figure 10: **Failure modes of TCoT** We show the precision and recall of frames that have been aggregated by our method, compared to the human-annotated time-reference frames of LVBench. This is performed for the instances in the dataset where TCoT fails, while the oracle which uses only the time-reference frames for answering succeeds.

being selected by the model. Overall accuracy degrades, particularly on LVBench, as discussed in Tab. 1.

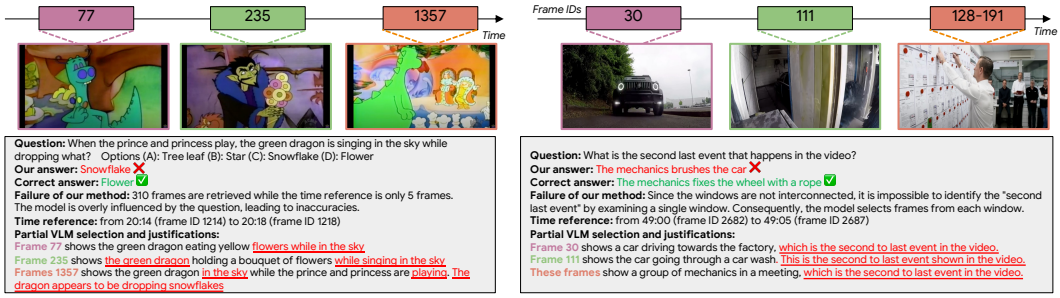
B Failure mode analysis

We examine the failure modes of our Temporal Chain of Thought in detail in Fig. 10. To do so, we use the annotated "time reference" segments from LVBench [58], which are segments of video that contain the necessary information to answer the question accurately. Ideally, assuming that the time-reference annotations are completely correct, we should only select the time-reference frames and no other ones.

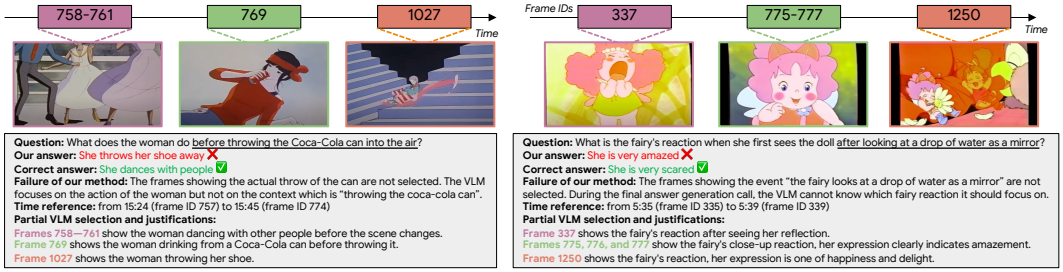
In Fig. 10 we evaluate the precision and recall of our selected frames, where a Frame ID is denoted as a "true positive" if it falls within the annotated time reference segment and "false positive" if it does not. High precision means that few selected frames fall outside the time reference frames, while high recall indicates that most of the time reference frames are included in the selection. In Fig. 10 we plot only the 168 instances where TCoT provides *incorrect* answers to the question, while the oracle answers correctly.

We identify prevalent failure modes in Fig. 10 and show qualitative examples for these modes in Fig. 11:

- **Low precision and high recall** (purple area in Fig. 10 and qualitative examples in Fig. 11a). The model selects too many frames, selecting frames that are only remotely related to the question. This over-eagerness leads to excessive and irrelevant frame selection. For example, we see in Fig. 11a (left) that if the question asks what the dragon is dropping while in the sky, the method selects frames of the dragon even when it is clearly not in the sky. Our model sometimes struggles with questions that require understanding connections between segments, such as determining the second-to-last event in a video as shown in Fig. 11a (right).



(a) **Failures due to low precision**



(b) **Failures due to low recall**

Figure 11: **Qualitative analysis of the failure modes of Temporal Chain of Thought. Failures due to low precision:** Sometimes, our method selects too many frames. In the top left example, it is overly influenced by the question and selects frames at each window, even when they are not highly relevant. For instance, if the question asks what the dragon is dropping while in the sky, the method retrieves frames of the dragon even when it is clearly not in the sky, incorrectly assuming it is. In the top right example, it is impossible to determine the "second last event" of the video by looking at a single window. As a result, the model incorrectly tries to answer the question within each single window. **Failures due to low recall:** Sometimes, our method misses important frames, such as those showing "throwing the Coca-Cola can" (left) or "looking at a drop of water as a mirror" (right). These missing frames are essential for accurate final inference.

- **Low recall and high precision** (orange area in Fig. 10 and qualitative examples in Fig. 11b): The selected frames are relevant, but the model misses crucial parts of the question. For example, in Fig. 11b (right) in response to the question, "What is the fairy's reaction when she first sees the doll after looking at a drop of water as a mirror?", TCoT selects frames of the fairy's reaction but omits frames showing the fairy looking at the water drop. Consequently, the final answer is inaccurate because the final VLM call cannot determine which of the fairy reactions it sees occurs after looking at a drop of water.

C Qualitative results

Figure 12 visualises additional examples of our TCoT method on LVBench [58], following the same format as Fig. 5 in the main paper.

Figure 13 shows an example of why adding uniform context (described in Sec. 3.2 of the main paper) is beneficial.

D Additional experimental details

Complete prompts Figure 3 in the main paper showed our VLM selection call, S (Eq. 4). For completeness, we also include the prompt for answering, H, (Eq. 3 of the main paper) too. Specifically, Fig. 14 and 15 shows the prompt for multiple-choice questions (Egoschema [41], LVBench [58] and NeXT-QA [66] datasets), and Fig. 16 for open-ended questions (for the OpenEQA [40] dataset).

Figure 12: **Additional qualitative examples on LVBench.** Note how our model focusses on different parts of the video to predict the correct answer. The top row shows an example of the model focussing on 3 diverse segments of the same video. The second row includes two such segments. The third row shows an example of how our TCoT approach is able to find the 4 frames in the entire video of 2075 frames which contain the correct answer. In all cases, for clarity, we show a single frame from each selected segment of frames. Faces have been blurred in the first row.

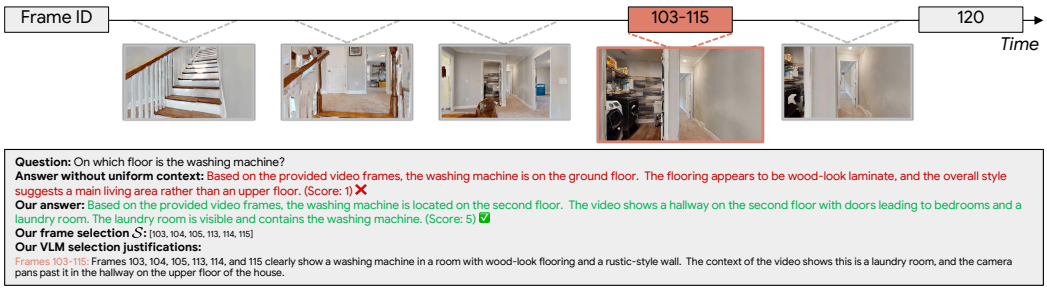


Figure 13: **Importance of adding uniform context** Our TCoT approach is able to select the relevant frames (103 to 115) for the question, by focussing on the washing machine (and provides the correct justification for doing so too). However, if we pass only these selected frames to the answerer, the result is incorrect as relevant information about which floor the machine is on is lost. To alleviate this issue, we therefore include uniformly sampled context as well, as visualised in this figure, and described in Sec. 3.2 of the paper. Example from OpenEQA [40].

```
You will be given a question about a video and {number of choices} possible answer
options. You are provided frames from the video, retrieved by an intelligent agent.

Frames: {frame1}, . . . , {frame N}
Question: {question}
Possible answer choices: {answer choices}

After explaining your reasoning, output the final answer in the format. "Final
Answer: (X)" where X is the correct digit choice. Never say "unknown" or "unsure", or
"None", instead provide your most likely guess.
```

Figure 14: **Multiple choice question prompt for the Gemini and GPT-4o-mini answering call**, H (Eq. 3).

```
Frames: {frame1}, . . . , {frame N}

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. Based on your
observations, select the best option that accurately addresses the question.
Question: {question}
Options: {answer choices}
Answer with the option's letter from the given choices directly and only give the best
option.
```

Figure 15: **Multiple choice question prompt for the Qwen 2.5-VL answering call**, H (Eq. 3).

Hierarchical TCoT We include further details of hierarchical aggregation, which was introduced in Sec. 4.2 of the main paper.

We hierarchically extend Single-Step TCoT (Sec. 3.2) by iteratively sampling around the previously identified frames of interest. Intuitively, in a long video where our model's context limit is far smaller than the number of input frames, if we first coarsely sample frames, we may miss many necessary frames. Therefore, once we have identified frames relevant to the question, we select nearby ones that were not considered initially and iterate.

Concretely, the first iteration follows "Single-Step TCoT" (Sec. 3.2): Given a set of frames, $\mathbf{x}_0$ sampled uniformly from the video, we identify relevant frames, $\hat{\mathbf{x}}_0, \mathcal{S}_0 = S(\mathbf{x}_0, \mathbf{q})$, where $\hat{\mathbf{x}}_0 = \{x_i, \ldots, x_j\}$ for $i, j \in \mathcal{S}_0$. Subsequently, we "zoom in" on these relevant regions by sampling additional frames around $\hat{\mathbf{x}}_0$ (Fig. 2b). Specifically, for each index in $\mathcal{S}_0$, we construct a neighbourhood of frames, $\mathbf{x}_1 = \{x_{i-v}, \ldots, x_i, \ldots, x_{i+v}, \ldots, x_{j-v}, \ldots, x_j, \ldots, x_{j+v}\}$, where v denotes the neighbourhood size, and duplicate frames are removed.

The new sequence of frames, $\mathbf{x}_1$, serves as the input for the next iteration. We repeat this algorithm until convergence, namely until t iterations have been completed or earlier if the relevant set of frames has not changed, or in other words, $\hat{\mathbf{x}}_{i+1} = \hat{\mathbf{x}}_i$.

Video Agent reimplementation We reimplemented the Video Agent framework [61] with several modifications to make it directly comparable to our approach. In particular, Video Agent used LaVila [73] as its captioner, whilst we use Gemini 1.5 Flash for this purpose for fair comparison. Similarly, instead of using EVA-CLIP-8b-plus [53], we use SigLIP-So400m/14 [71] as the other baselines in Tab. 2. Finally, we used Gemini 1.5 Flash as the answerer to be consistent with the rest of our work, instead of the original GPT-4. Video Agent starts off with an initial number of uniformly selected frames. We ablated this hyperparameter and achieved the best performance with 50 initial frames on Egoschema, and 64 initial frames on LVBench.

```
Frames: {frame1}, . . . , {frame N}

You will be given a question about a video.You will be provided frames from the
video, retrieved by an intelligent agent. It is crucial that you imagine the visual
scene as vividly as possible to enhance the accuracy of your response.
Question: {question}
```

Figure 16: **Open-ended question prompt for our Gemini answering call**, H (Eq. 3).

Details of captioning for feature-similarity based aggregation An additional baseline that we used in the paper was retrieve frames based on the feature-similarity to their captions (Tab. 2). Here, we include additional details of it.

We generate two types of captions for each frame of the video frame using Gemini 1.5 Flash. For *concise* captions, we use the prompt *Write a concise description of the image in a sentence*. For *long* captions, we use the prompt *Write a paragraph describing the image in detail*. These captions were then embedded using the most performant version of SigLIP [71], SigLIP-So400m/14. We then used these embeddings to perform nearest-neighbour search, selecting the most similar captions to the input questions. The frames corresponding to these captions were then used to aggregate the context, $\mathbf{c}$, that was then passed to the answerer VLM.

Computational Resources We call the Gemini 1.5 Flash [55] and GPT-4o-mini [1] through their public APIs. Qwen-2.5-VL [6] has publicly-available weights, and we run the HuggingFace implementation using a server with 8x NVIDIA A100 GPUs.

Video IDs from LVBench The videos in LVBench [58] are originally from YouTube. Since some of these videos are no longer available, we include the full list of Video IDs that we were able to obtain below.

Each video has on average 15.1 questions associated with it, for a total of 1043 questions.

The full list of all Video IDs that we were able to obtain are:

- `-WnyRMZqV1U`
- `16Z-XQh9jhk`
- `2LH3JCGkEBU`
- `2sriHX3PbXw`
- `2zkJFv-ro4A`
- `3_upA09AntU`
- `4LA_tH-VSnQ`
- `81SbCR6p3Z0`
- `9-gOCOu_KGU`
- `9tBsMSDoDqk`
- `AeEYQ62t8hA`
- `CgvJqGxzRfE`
- `Cm73ma6Ibcs`
- `EwskdNETNx8`
- `FaV0tIaWWEg`
- `HfEVEGf1A8Q`
- `JPPMz8fEml0`
- `JTa_Ue2MSwc`
- `JlrzSvCsIjE`
- `KbahC-QCKU8`
- `Mcggugol2ts`
- `NzCO0G8AGLU`
- `O14bbpvy2x0`
- `QB7FoIpx8os`
- `QgWRyDV9Ozs`
- `RCAqKnvu_P0`

- RjDrZkBwzho
- S8vPx-u9p_A
- SRq0weUKskM
- TJR1oYDDTwg
- TZ0j6kr4ZJ0
- VTCDQYYKA9o
- Va_9Q6ekm60
- Vk_Af0htZGU
- W-BnDvXXfOs
- XNtNNplAwiI
- Xjf5N9S3jAA
- YlQugR7KSKg
- Z4HGQL_McDQ
- Z86xysw5Ncc
- Za2Z_JRxCuk
- _T2Avd3tFHc
- aJI8XTa_DII
- cWEnogdsW78
- cXDT44zT8JY
- gbDR39yIs3Y
- hROKtPqktO8
- hjoDzK0siaM
- iA_69g87Ilw
- ihfjEFGdZdc
- jp2M1hIEtsk
- k2FIFQIYBvA
- lDlA7cfNk8A
- o-gLbgpzCc8
- pXD3txG2bVQ
- q01CUy_gwdU
- qAIRFyR6NyQ
- qYMnM5blZIE
- rk24OUu_kJQ
- rp4NKWb7dXk
- sk00epALZps
- t-RtDI2RWQs
- tH_5YbklevQ
- tKIFQI9cH2c
- uW9mcG0rdLY
- vHlSoxg8WHo
- vaL_vSdZKZo
- wgBlACG927Y
- xi6r3hZe5Tg