Mitigating Hallucination in VideoLLMs via Temporal-Aware Activation Engineering

Jianfeng Cai[†] **Jiale Hong**[‡] **Zongmeng Zhang**[†]

Wengang Zhou[†*] **Nianji Zhan**[§] **Houqiang Li**[†]

[†]University of Science and Technology of China [‡]Shanghai Jiaotong University
[§]Merchants Union Consumer Finance Company Limited
xiaobaicai@mail.ustc.edu.cn

Abstract

Multimodal large language models (MLLMs) have achieved remarkable progress in video understanding. However, hallucination, where the model generates plausible yet incorrect outputs, persists as a significant and under-addressed challenge in the video domain. Among existing solutions, activation engineering has proven successful in mitigating hallucinations in LLMs and ImageLLMs, yet its applicability to VideoLLMs remains largely unexplored. In this work, we are the first to systematically investigate the effectiveness and underlying mechanisms of activation engineering for mitigating hallucinations in VideoLLMs. We initially conduct an investigation of the key factors affecting the performance of activation engineering and find that a model's sensitivity to hallucination depends on **temporal variation** rather than task type. Moreover, selecting appropriate internal modules and dataset for activation engineering is critical for reducing hallucination. Guided by these findings, we propose a temporal-aware activation engineering framework for VideoLLMs, which adaptively identifies and manipulates hallucination-sensitive modules based on the temporal variation characteristic, substantially mitigating hallucinations without additional LLM fine-tuning. Experiments across multiple models and benchmarks demonstrate that our method markedly reduces hallucination in VideoLLMs, thereby validating the robustness of our findings[2].

1 Introduction

Large language models (LLMs) [1, 27, 48, 72, 3] have witnessed explosive growth in recent years, revolutionizing the field of artificial intelligence with remarkable capabilities in understanding and generating human language. This success has naturally extended to multimodal domains, where LLMs are adapted to process and reason about diverse data types beyond text [19, 53, 32, 64, 84]. Particularly noteworthy is the emergence of Multimodal LLMs (MLLMs) in the image domain (ImageLLMs) [40, 31, 49, 5] that integrate visual and spatial perception with language understanding, and in the video domain (VideoLLMs) [6, 80, 47, 53], which further capture temporal dynamics to enable comprehensive video understanding through joint spatiotemporal modeling.

Despite their impressive performance, MLLMs inherit and even amplify the hallucination issues prevalent in LLMs [7, 76, 78, 60]. ImageLLMs frequently hallucinate non-existent objects [59],

*Corresponding author.

[2]Code and dataset are available at https://github.com/cai-jianfeng/TA-AE and https://huggingface.co/datasets/caijanfeng/TA-AE

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

misidentify visual elements [28, 39], or fabricate spatial relationships [70]. In the video domain, the additional temporal dimension substantially exacerbates hallucination problems [52]. VideoLLMs often confabulate events that never appear in the footage [82], misinterpret action sequences [41], or fail to maintain temporal consistency [17, 69], which creates increasingly complex challenges.

To alleviate the hallucination issues in VideoLLMs, researchers have proposed various approaches that broadly fall into two categories: model fine-tuning methods [52, 4, 66, 38] and train-free methods [63, 79, 41, 82]. Model fine-tuning approaches necessitate additional training, either by training on carefully curated datasets [73] or by training auxiliary modules [52] specifically designed to mitigate hallucination. However, these methods face significant challenges, notably the labor-intensive collection of high-quality datasets and the heavy computational cost of training large models. Consequently, train-free methods have emerged as alternative solutions, encompassing self-feedback mechanisms [79], contrastive decoding strategies [82], and techniques that leverage auxiliary information [41]. Such train-free approaches offer the advantage of improving hallucination mitigation without necessitating costly retraining or extensive data collection efforts.

Activation engineering [50, 85, 44, 9] is an emerging techniques that manipulate model activations during inference to influence model behavior. This approach has been widely applied in domains such as truthfulness control [45] and sentiment modulation [65, 36]. Recently, researchers have begun to explore its application in mitigating hallucinations in LLMs [45] and ImageLLMs [12, 51]. These works indicate that activation engineering can better leverage correlations, reduce hallucinations without introducing additional side effects, while requiring fewer extra resources. However, its feasibility in the context of VideoLLMs remains an open question.

In this work, we introduce activation engineering to mitigate hallucinations in VideoLLMs for the first time and conduct a systematic investigation into its feasibility and underlying mechanisms, aiming to identify the key factors that influence its effectiveness. Our findings demonstrate that the two variants of activation engineering exhibit similar performance under any given dataset, indicating no essential difference between them. Furthermore, we reveal that the hallucination sensitivity of internal modules in VideoLLMs is strongly correlated with the temporal variation characteristics of tasks, rather than the task type. Moreover, selecting appropriate modules and datasets for activation engineering is crucial for effectively mitigating hallucinations.

Based on these findings, we propose a temporal-aware activation engineering framework that adaptively identifies and manipulates hallucination-sensitive modules according to temporal variation characteristics. Specifically, we divide the temporal variation characteristics of videos into temporal-invariant and temporal-variant, and design a fully automated pipeline to collect high-quality datasets for each characteristic. Next, we train a lightweight classifier to predict the temporal variation of a given input. Guided by the classifier, we select the corresponding dataset to identify hallucination-sensitive modules in the VideoLLM and apply activation engineering to mitigate hallucinations. Our experiments demonstrate the effectiveness of our method, yielding consistent gains across multiple benchmarks (e.g., up to $+5.52\%$ on VidHalluc [41] and up to $+24.21\%$ on EventHallusion [82]) and across diverse models, including Video-LLaVA [47], VideoLLaMA2 [16], and Qwen2.5-VL [6].

In summary, our primary contributions are as follows:

- We are the first to introduce activation engineering to mitigate hallucinations in VideoLLMs and conduct a systematic investigation into its feasibility and intrinsic mechanisms.
- Our study reveals that the hallucination sensitivity of internal modules in VideoLLMs is strongly correlated with temporal variation, e.g., temporal changes within videos.
- We propose an effective activation engineering framework, which employs a fully automated dataset construction pipeline and a temporal variation classifier to guide module selection.
- Extensive experiments across multiple models and benchmarks demonstrate that our method significantly reduces hallucinations without any additional LLM training, while validating the correctness of our findings.

2 Related Work

In this section, we briefly introduce the background of MLLMs, hallucinations in MLLMs, and Activation Engineering. A more detailed version is provided in Appendix A.

2.1 MLLMs for Video Understanding

Following the success of Large Language Models (LLMs) [8, 18, 26, 48], researchers have extended them to incorporate visual inputs, giving rise to multimodal LLMs (MLLMs)[2, 14, 43]. In the video domain, frames are typically divided into patches treated as tokens[21], then processed by a temporally visual encoder and aligned with the LLM token space via a non-linear module [47, 6, 80]. For example, Video-LLaVA [47] uses time-sensitive attention to capture frame-wise dynamics; the VideoLLaMA series [81, 16, 80] leverages Q-formers [42] or convolutional downsampling to extract temporal features while reducing token count; and Qwen2.5-VL [6] applies dynamic frame rate training with absolute time encoding to support resolution and frame rate adaptation.

2.2 Hallucinations in MLLMs

Despite their strong performance, MLLMs often suffer from hallucinations [60, 78, 7], generating inaccurate or fabricated responses due to misinterpretation of input data. To address this in the image domain, two main strategies have emerged: training-based and training-free methods. The former improves alignment through additional data [34, 35, 24], but incurs high computational costs. The latter mitigates hallucinations during inference without retraining [74, 10, 67]. For instance, Volcano [38] use self-feedback to refine outputs, while VCD [39] apply contrastive decoding [46] to suppress hallucinated tokens. Other approaches include hallucination detection and correction [15, 29], and boosting visual grounding via auxiliary signals like attention maps [83, 33, 11].

When extending MLLMs to the video domain, hallucination becomes more challenging due to temporal dynamics [52, 69]. To address this, researchers have improved image-based techniques, such as collecting high-quality datasets for fine-tuning [52, 38], applying self-feedback [79], leveraging contrastive decoding [82], correcting hallucinations [22], and incorporating auxiliary signals to enhance video understanding [41]. For example, Vript [73] retrains the model on a $12k$-sample dataset to reduce hallucinations; TCD [82] uses contrastive decoding to suppress hallucinated tokens; and DINO-HEAL [41] re-weights video embeddings using attention maps from a DINO [54] model. Concurrently, benchmarks like VidHal [17], EventHallusion [82], and VidHalluc [41] have been proposed to evaluate hallucination across different aspects. Despite these efforts, hallucination in VideoLLMs remains relatively under-explored compared to the image domain.

2.3 Activation Engineering

Activation engineering [65, 85, 44, 30, 58] refers to manipulating the activations within a model to control its behavior. Prior works have demonstrated its effectiveness on truthfulness [45], formality transfer [50], sentiment control [65, 36] and refusal behavior control [9].

ITI [45] was the first to apply this technique to mitigate textual hallucinations in LLMs by computing activation vectors from normal samples and hallucination-inducing samples, and injecting them into the multi-head attention layers during inference to suppress hallucinated content. More recently, several studies have extended activation engineering to address hallucinations in MLLMs for image-related tasks [51, 12]. Specifically, ICT [12] computes image-level and object-level vectors to reduce both global and local hallucinations. VTI [51] builds upon this by injecting activation vectors into both the vision encoder layers and the LLM layers, thereby mitigating hallucinations more effectively.

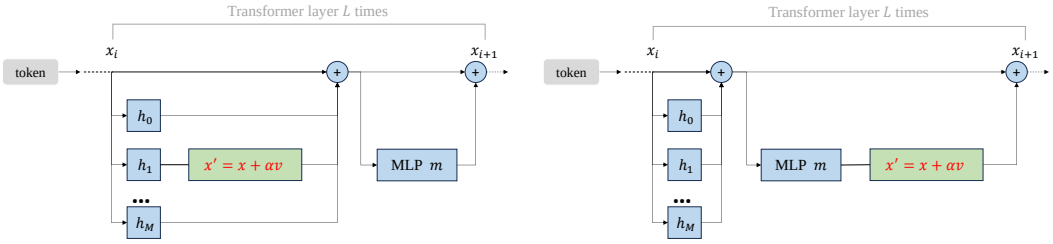
However, the feasibility of applying it to VideoLLMs remains an open question. In this paper, we explore the applicability of activation engineering for mitigating hallucinations in VideoLLMs.

3 Temporal Variation Matters in Activation Engineering

In this section, we investigate the internal mechanisms of activation engineering and how key factors influence its effectiveness in mitigating VideoLLM hallucinations under varying conditions.

3.1 Preliminary

Activation Engineering. Activation engineering mitigates hallucinations by computing offset vectors between normal and hallucination-inducing prompts, then injecting these averaged vectors into specific modules in the model during inference to guide the model toward non-hallucinated outputs.



(a) Attention Head Engineering (b) Layer Engineering

Figure 1: Overview of two common activation engineering variants.

To enable it, we need to construct a dataset $\mathcal{D} = \{(x_n^{(i)}, y^{(i)}), (x_h^{(i)}, y^{(i)})\}_{i=1}^{|\mathcal{D}|}$, where $x_n^{(i)}$ and $x_h^{(i)}$ are normal and hallucination-inducing prompts, respectively, paired with a shared ground-truth response $y^{(i)}$. Each pair is fed into the model, and the vectors at the final token position are extracted from N modules, yielding $v_n^{(i,j)}$ and $v_h^{(i,j)}$ for module $j = \{1, ..., N\}$. We define the offset vector $v^{(i,j)} = v_n^{(i,j)} - v_h^{(i,j)}$ to capture the latent deviation induced by hallucinations. Aggregating across the dataset gives $\mathcal{V}^{(j)} = \{v^{(i,j)}\}$, after which an average offset vector $v^{(j)} = \sum_{i=1}^{|\mathcal{D}|} v^{(i,j)}/|\mathcal{D}|$ is computed per module and later injected during inference to mitigate hallucinations.

As shown in Figure 1, activation engineering is divided into Attention Head Engineering and Layer Engineering, based on the target modules. The former operates at the attention head level, and the latter operates at the transformer layer level. Taking Attention Head Engineering as an example, for an LLM with L layers and M heads per layer, $L \times M$ offset vectors can be computed. Prior work [45] suggests only a subset of heads are strongly hallucination-sensitive; thus, instead of using all heads, we selectively focus on the most sensitive ones to reduce computation.

To identify these, we train a binary classifier for each attention head j using its vector set $\mathcal{V}_{nh}^{(j)} = \{v_n^{(i,j)}, v_h^{(i,j)}\}$. The top-$K$ heads with the highest classification accuracy are retained, and offset vectors are computed and applied only to these during inference. See Appendix B for more details.

Temporal Variation Characteristic [61, 86]. A video is considered to exhibit temporal-invariant characteristic when it primarily involves fine-grained events without significant scene changes or temporal dynamics. Conversely, a video is regarded as having temporal-variant characteristic when it contains multiple events with substantial scene transitions and temporal variations.

Evaluations. Due to its diversity and comprehensive coverage, we adopt the VidHalluc [41] benchmark as our evaluation benchmark. Within VidHalluc, BQA and MCQ emphasize fine-grained action comprehension and thus possess temporal-invariant characteristic, while STH and TSH target temporal scene transitions and thus possess temporal-variant characteristic. Additionally, these tasks span distinct task types: BQA (binary judgment), MCQ (multiple choice), STH (question answering), and TSH (ranking). We adopt VideoLLaMA2 [41] as the base model for evaluation.

3.2 Frame Reduction Induces Hallucination

Activation engineering requires a collection of normal prompts and hallucination-inducing prompts to compute activation vectors. Following TCD [82], for a given sample (m, q, y), where m denotes the input video, q represents the question, and y is the corresponding response, the normal prompt x_n is constructed by concatenating the video and the question, e.g., $x_n = (m, q)$. To generate a hallucination-inducing prompt x_h, we apply the frame downsampling technique to the video m and use the downsampled video m_d combined with the same question q as x_h, e.g. $x_h = (m_d, q)$.

To validate the effectiveness of this strategy, we evaluate VideoLLaMA2 on the VidHalluc under different frames. As

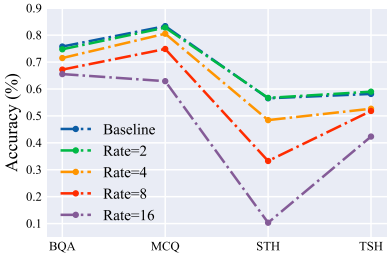
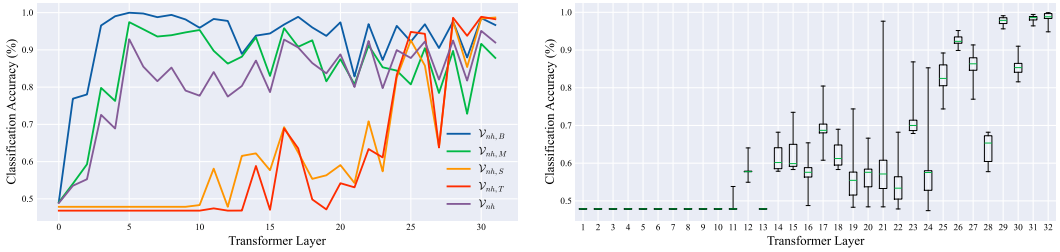


Figure 2: Frame reduction amplifies hallucinations in VideoLLM. *Rate* denotes the frame downsampling rate.

shown in Figure 2[3], the model's performance consistently degrades as the number of frames decreases, indicating an increased degree of hallucination. This supports the validity of using frame-downsampled videos as hallucination-inducing prompts.

3.3 Hallucination Sensitivity Correlates with Temporal Variation Not Task Type



(a) Layer-wise mean binary-classifier accuracy across attention heads trained on vector sets $\mathcal{V}_{nh,*}$.

(b) Layer-wise distribution of binary-classifier accuracy across attention heads trained on $\mathcal{V}_{nh,S}$.

Figure 3: Binary-classifier performance trained on vector sets $\mathcal{V}_{nh,*}$ from different task types. (b) shows results for $\mathcal{V}_{nh,S}$ with the remaining results presented in Figures 8 and 9 of Appendix F.

To further investigate, we take Attention Head Engineering as a case study. Specifically, we sample a fixed number of examples from each of BQA, MCQ, STH, and TSH, and construct corresponding datasets $\mathcal{D}_* = (m_*, q_*, y_*)$, where $* \in \{B, M, S, T\}$. For each example, we use the original video and question as the normal prompt $x_{n,*} = (m_*, q_*)$, and construct a hallucination-inducing prompt $x_{h,*} = (m'_*, q_*)$, where m'_* is obtained by downsampling the original video frames by a factor of 4.

We then use $\mathcal{D}_*$ to extract the vectors from each attention head in VideoLLaMA2 and obtain the vector set $\mathcal{V}_{nh,*}^{(j)} = \{v_{n,*}^{(i,j)}, v_{h,*}^{(i,j)}\}$ for every head j. Each vector set $\mathcal{V}_{nh,*}^{(j)}$ is split into a training set and a validation set. For each attention head j, we train a binary classifier using the training set to determine whether the input vector v originates from a hallucination-inducing prompt. Finally, we evaluate each classifier on the corresponding validation set. The results of binary classifiers individually trained on each of the four vector sets $\mathcal{V}_{nh,*}$ are shown in Figures 3, 8, and 9(a).

We observe two key findings from the results. First, based on the results shown in Figures 3(b), 8, and 9(a), for all tasks types, the attention heads within each transformer layer exhibit highly similar performance, That is, the classification accuracy of the M attention heads within the same layer is nearly identical. This suggests that **the attention heads within a transformer layer share a similar capacity for perceiving hallucinations**, which contrasts with prior studies [45].

Second, as illustrated in Figure 3(a), for tasks that have the same temporal variation characteristics but different task types, such as BQA and MCQ which possess temporal-invariant characteristic, the attention heads demonstrate similar capabilities in hallucination detection. Specifically, the accuracy trends of their corresponding classifiers are consistent. This indicates that **the internal modules' hallucination sensitivity is task type agnostic**, meaning their ability to perceive hallucinations does not depend on the task type (e.g., binary classification or multiple-choice). Instead, **these modules' hallucination sensitivity is correlated with the temporal variation characteristics of tasks**, indicating that internal modules of the model (e.g., attention heads or layers) that effectively detect hallucinations vary according to the temporal variation characteristics of tasks. We also provide an intuitive explanation of this phenomenon in Appendix C.

3.4 Module Selection Determines Activation Engineering Performance

Next, we evaluate the Attention Head Engineering on the VidHalluc using each dataset $\mathcal{D}_*$. Specifically, we select the top-K attention heads with the highest classification accuracy based on the binary classifiers introduced in Section 3.3. For each head $j_k, k = \{1, 2, ..., K\}$, we compute the offset vector set $\mathcal{V}_*^{(j_k)}$ using the corresponding vector set $\mathcal{V}_{nh,*}^{(j_k)}$. During inference with the VideoLLM, we apply activation engineering to the selected K attention heads. The results are presented in Table 1.

[3]Unless otherwise specified, Baseline in this paper refers to direct evaluation of the VideoLLM.

Table 1: Results of Attention Head Engineering and Layer Engineering on VidHalluc using different datasets. "Attn Head" denotes Attention Head Engineering, while "Layer" denotes Layer Engineering.

Datasets	BQA		MCQ		STH		TSH	
	Attn Head	Layer	Attn Head	Layer	Attn Head	Layer	Attn Head	Layer
Baseline	75.77	75.77	83.55	83.55	56.55	56.55	58.17	58.17
$\mathcal{D}_B$	76.82	74.77	83.70	83.33	54.28	49.55	56.83	55.33
$\mathcal{D}_M$	78.29	76.07	83.77	83.70	54.38	46.59	56.67	54.83
$\mathcal{D}_S$	77.22	76.13	83.89	83.75	66.37	60.52	59.00	58.83
$\mathcal{D}_T$	79.45	81.57	83.49	83.58	64.88	66.38	58.50	57.83

It can be observed that when Attention Head Engineering is performed using datasets with the same temporal variation characteristics as the task, such as using $\mathcal{D}_B$ for the MCQ task, it consistently produces positive effects. In contrast, when using datasets that have different temporal variation characteristics than the task, the effectiveness becomes inconsistent. Specifically, compared to the Baseline, using $\mathcal{D}_B$ or $\mathcal{D}_M$ for the STH or TSH tasks leads to a decline in performance, whereas applying $\mathcal{D}_S$ or $\mathcal{D}_T$ for the BQA or MCQ tasks results in improved performance.

Further examination of Figure 3(a) reveals an asymmetry in hallucination sensitivity across tasks with different temporal variation characteristics. The attention heads that are sensitive to hallucinations in the STH or TSH tasks also exhibit sensitivity to hallucinations in the BQA or MCQ tasks. However, the attention heads that are sensitive in BQA or MCQ tasks do not exhibit the same sensitivity in STH or TSH tasks. Therefore, the offset vector sets $\mathcal{V}_S$ and $\mathcal{V}_T$ remain effective when applied to BQA and MCQ tasks, while the $\mathcal{V}_B$ and $\mathcal{V}_M$ are ineffective for STH and TSH tasks.

To further validate this hypothesis, we apply Attention Head Engineering to the STH and TSH tasks using the selected attention heads $\mathcal{A}_S$ from $\mathcal{D}_S$ and the corresponding offset vector sets $\mathcal{V}_B$ and $\mathcal{V}_M$. The results are presented in Table 2. It can be observed that Attention Head Engineering continues to yield positive effects. This indicates that the offset vector sets computed from attention heads with strong hallucination sensitivity across tasks with different temporal variation characteristics, such as those in the deeper layers shown in Figure 3(a), can be used interchangeably. In contrast, offset vector sets derived from attention heads with temporal variation specific hallucination sensitivity, such as those in the shallow layers shown in Figure 3(a), cannot be used across tasks with different temporal variation characteristics. Therefore, **carefully selecting appropriate attention heads for each task based on its temporal variation characteristics** is essential for the effectiveness of activation engineering.

Table 2: Results of Attention Head Engineering on STH and TSH tasks using different dataset and selected attention heads combinations.

Datasets	STH	TSH
Baseline	56.55	58.17
$\mathcal{D}_B + \mathcal{A}_S$	59.37	58.50
$\mathcal{D}_M + \mathcal{A}_S$	65.98	59.33

3.5 Attention Head Engineering and Layer Engineering Exhibit Similar Trends

Next, we evaluate the Layer Engineering on the VidHalluc using each dataset $\mathcal{D}_*$. The results are shown in Table 1. It can be observed that the performance of Layer Engineering closely resembles that of Attention Head Engineering. Furthermore, as shown in Section 3.3, attention heads within the same layer exhibit similar sensitivity to hallucinations. This suggests that **Attention Head Engineering and Layer Engineering may be intrinsically equivalent**, as both methods achieve consistent effects when applied during model inference using the same dataset.

3.6 Mixed Dataset Produces Moderate Effects Across All Tasks

Finally, we combine the four datasets $\mathcal{D}_*$ into a mixed dataset $\mathcal{D}$, and evaluate the Attention Head Engineering on the VidHalluc using $\mathcal{D}$. As shown in Table 3, the mixed dataset leads to consistent performance improvements across all four tasks. However, the magnitude of improvement is smaller compared to the results reported in Table 1.

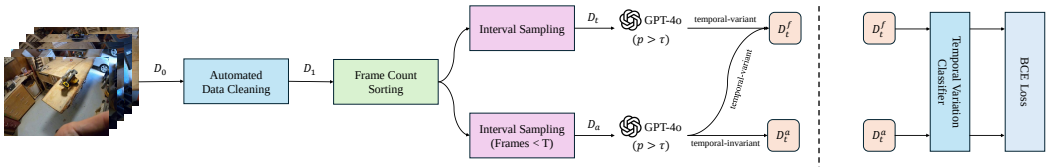
Table 3: Results of Attention Head Engineering on VidHalluc using mixed dataset.

Datasets	BQA	MCQ	STH	TSH
Baseline	75.77	83.55	56.55	58.17
$\mathcal{D}$	76.02	83.58	59.46	58.33

We further visualize the classification performance of each attention head based on the mixed dataset, as indicated by the purple curve (labeled $\mathcal{V}_{nh}$) in Figure 3(a). It can be observed that the attention heads identified as sensitive to hallucinations from $\mathcal{D}$ partially overlap with those selected from $\mathcal{D}_B$ and $\mathcal{D}_M$, and partially with those from $\mathcal{D}_S$ and $\mathcal{D}_T$. As a result, when applying activation engineering, only a subset of the selected attention heads aligns with those effective for each specific task. This partial mismatch ultimately leads to less significant improvements compared to those in Table 1. This indicates that **naïvely mixing dataset from tasks with different temporal variation characteristics does not lead to further performance gains across tasks**. Instead, compared to using task-specific datasets, it results in a trade-off that balances performance between tasks.

Consequently, it is necessary to construct separate datasets for tasks with different temporal variation characteristics to enable the selection of effective modules and the computation of corresponding offset vector sets to fully leverage the potential of activation engineering.

4 Temporal-Aware Activation Engineering: Method and Experiments



(a) Automated Dataset Collection Pipeline for Diverse Tasks (b) Classifier Training Framework

Figure 4: Automated dataset collection pipeline and temporal variation classifier training framework.

In this section, we construct temporal-aware activation engineering framework based on the findings of Section 3. The overall framework is illustrated in Figure 4. We then evaluate our method through comprehensive experiments across various models and benchmarks, demonstrating its effectiveness in reducing hallucinations, thereby substantiating our findings.

4.1 Temporal-Aware Activation Engineering Framework

Basic Task Dataset Processing. Unlike earlier studies [12, 51] that draw only from benchmark subsets, we leverage an existing open-source VQA dataset to preserve data diversity while preventing the performance inflation that arise from benchmark leakage. Specifically, we refrain from reusing the benchmark sources employed in Section 4.2 and instead adopt the ShareGPT4Video dataset [13] as our base collection, denoted as $\mathcal{D}_0 = \{(m_i, q_i, y_i)\}_{i=1}^{N}$. According to Section 3.3, hallucination sensitivity of the model's internal modules is task type agnostic. Therefore, we directly keep the original video description as the task type in our dataset.

Consistent with the finding in Section 3.3, the decisive factor for the hallucination perception capability of modules is the temporal variation characteristics of tasks. We therefore divide video understanding tasks into two sub-tasks: the task with temporal-invariant characteristic and the task with temporal-variant characteristic, and we prepare distinct datasets for each task.

Since activation engineering requires only a small amount of high-quality data, we clean $\mathcal{D}_0$ to obtain approximately $10k$ samples in $\mathcal{D}_1$, with details in Appendix D.1. Then the videos in $\mathcal{D}_1$ are then sorted by frame count. To capture a wide temporal span without processing the entire $\mathcal{D}_1$, we select samples at uniform intervals, yielding $1k$ videos that form the candidate pool for the temporal-variant task, denoted $\mathcal{D}_t$. For the temporal-invariant task, whose videos are typically short, we discard any video in $\mathcal{D}_2$ whose frame count exceeds a threshold T, and again apply uniform-interval sampling to obtain $1k$ candidates, resulting in the dataset $\mathcal{D}_a$. In our experiments, T is set to 200.

Filtering Datasets for Distinct Tasks. The two candidate datasets $\mathcal{D}_a$ and $\mathcal{D}_f$ undergo an additional refinement stage. Guided by the prompt illustrated in Figure 7 in Appendix D.2, GPT-4o [32] evaluates every sample in $\mathcal{D}_a$ and $\mathcal{D}_t$, judging whether its characteristic is temporal-invariant or temporal-variant. Any entry for which the model assigns a Yes/No confidence below τ is removed to eliminate ambiguous cases. In $\mathcal{D}_a$, only samples classified as temporal-invariant are preserved, yielding $\mathcal{D}_a^f$. In contrast, $\mathcal{D}_t$ retains solely the temporal-variant samples; the temporal-variant items

that are excluded from $\mathcal{D}_a$ are also merged into this set, producing $\mathcal{D}_t^f$. Statistics regarding $\mathcal{D}_a^f$ and $\mathcal{D}_t^f$ can be found in Appendix D.3.

Temporal Variation Classifier Training. To determine the temporal variation characteristics of a given (m, q) pair, we train a temporal variation classifier θ on $\mathcal{D}_a^f$ and $\mathcal{D}_t^f$. We randomly sample 400 instances from each dataset as the training set and use the remaining samples for validation. Each (m, q) is fed into the classifier, which is optimized with a binary cross-entropy loss. The model that attains the highest validation accuracy is adopted as the final classifier.

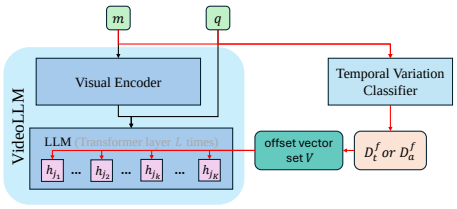


Figure 5: Inference process of the temporal-aware activation engineering framework.

To avoid additional latency during VideoLLM inference, the classifier's execution is overlapped with the model's forward pass. Specifically, (m, q) is supplied in parallel to both the VideoLLM visual encoder and classifier θ. Once the classifier returns classification result, the corresponding pre-computed offset vector set from either $\mathcal{D}_a^f$ or $\mathcal{D}_t^f$ is selected and injected into the LLM to continue the inference process. Because these offset vector sets are computed only once per VideoLLM, no extra runtime overhead is introduced. The complete inference process is illustrated in Figure 5.

4.2 Experimental Settings

Benchmarks. We assess the effectiveness of our method on two representative benchmarks. (a) VidHalluc [41] is a comprehensive benchmark for measuring hallucination in VideoLLMs. It covers three dimensions, action, temporal sequence, and scene transition, and comprises four task types: BQA, MCQ, STH, and TSH. For STH, the evaluation metric is an overall score obtained by a weighted combination of binary-classification task and the description task; the other three subtasks are evaluated purely by the accuracy of their respective tasks. (b) EventHallusion [82] is a recently proposed benchmark targeting hallucinations about events. It contains three subtasks, Entire, Mix, and Misleading, whose evaluation metric is binary-classification accuracy.

Models and Baselines. We adopt three VideoLLMs, Video-LLaVA [47], VideoLLaMA2 [16], and Qwen2.5-VL [6], as baseline models. For comparison, we include two inference-time hallucination-mitigation methods, TCD [82] and DINO-HEAL [41]. TCD applies contrastive decoding to VideoLLMs, whereas DINO-HEAL re-weights video features using attention weights produced by DINO. In addition, we employ CGD [20] and VTI [51], two representative contrastive decoding methods in the image domain, as comparative methods for evaluation on the VidHalluc benchmark. Since both CGD and VTI were originally proposed for image-based scenarios, we extend them to the video domain with several adaptations. Detailed implementation procedures are provided in Appendix E.

Implementation details. Unless explicitly stated otherwise, all experiments employ Attention Head Engineering as our primary method. For each VideoLLM, we pre-compute the Top-K attention-heads and the corresponding offset vector sets from $\mathcal{D}_a$ and $\mathcal{D}_t$. To ensure reproducibility, all methods use greedy decoding. Hyperparameters are tuned via grid search to systematically explore critical settings. Additional experimental details are provided in the Appendix E.

4.3 Experimental Results and Analysis

Results on VidHalluc. Table 4 presents the performance on VidHalluc. A noteworthy finding is the robustness of our approach: it delivers substantial improvements over the baseline models across nearly all subtasks and models. For VideoLLaMA2, the gains on the MCQ, STH, and TSH subtasks are considerably larger than those obtained when the model is supplied with each subtask's original dataset (Table 1), clearly demonstrating the value of our dataset. Relative to the compared mitigation methods, our approach achieves the best *Overall* performance on all models and attains the highest results in almost every subtask, with most margins exceeding 5%. In particular, on the *Overall* metric, our method surpasses TCD by up to 6.15%, DINO-HEAL by up to 4.5%, CGD by up to 3.88%, and VTI by up to 2.16%. On BQA and MCQ tasks, the maximum improvements reach 2.32% and 0.87% over TCD, 6.41% and 9.26% over DINO-HEAL, 3.06% and 9.77% over CGD, and 0.95% and 5.13% over VTI respectively. While on STH and TSH tasks, the corresponding gains

Table 4: Results on VidHalluc benchmark.

Model	Variant	BQA	MCQ	STH	TSH	Overall
VideoLLaMA2	Baseline	75.77	83.35	56.55	58.17	68.46
	TCD	76.77	83.65	55.19	56.67	68.07
	DINO-HEAL	75.79	83.35	56.32	57.67	68.28
	CGD	76.03	82.76	59.14	59.02	69.24
	VTI	78.14	83.67	62.47	60.28	71.14
	Ours	**79.09**	**84.07**	**65.14**	**61.67**	**72.49**
Qwen2.5-VL	Baseline	73.50	63.66	60.55	59.17	64.22
	TCD	75.49	74.35	46.81	**65.83**	65.62
	DINO-HEAL	69.77	65.69	60.97	59.83	64.07
	CGD	74.23	65.18	60.97	59.86	65.06
	VTI	75.49	69.82	61.74	60.97	67.01
	Ours	**76.18**	**74.95**	**61.81**	61.33	**68.57**
Video-LLaVA	Baseline	67.75	66.60	21.80	46.83	50.75
	TCD	70.40	65.97	21.77	42.33	50.12
	DINO-HEAL	**70.82**	**67.19**	26.47	47.50	53.00
	CGD	67.42	65.79	29.17	47.29	52.42
	VTI	68.23	66.14	33.24	48.96	54.14
	Ours	67.70	66.84	**41.16**	**49.50**	**56.30**

are up to 19.39% and 7.17% over TCD, 14.69% and 4.00% over DINO-HEAL, 11.99% and 2.67% over CGD, and 7.92% and 1.39% over VTI. These consistent improvements over both video-native (TCD, DINO-HEAL) and image-originated (CGD, VTI) methods highlight the effectiveness and generalizability of our approach.

Table 5: Results on EventHallusion benchmark.

Model	Variant	Entire	Mix	Misleading	Overall
VideoLLaMA2	Baseline	38.60	62.18	46.08	51.59
	TCD	42.98	**75.65**	54.90	**61.37**
	DINO-HEAL	38.60	62.69	46.08	51.83
	Ours	**43.86**	70.47	**57.70**	59.17
Qwen2.5-VL	Baseline	71.93	39.38	98.04	63.09
	TCD	69.30	**43.01**	97.06	63.81
	DINO-HEAL	80.70	35.75	95.10	63.08
	Ours	**89.47**	40.93	**100.00**	**69.19**
Video-LLaVA	Baseline	41.23	37.82	69.61	46.70
	TCD	46.49	54.92	79.41	58.68
	DINO-HEAL	39.47	48.71	79.41	53.79
	Ours	**79.83**	**55.96**	**90.20**	**70.91**

Results on EventHallusion. EventHallusion results are shown in Table 5, indicating that our method yields consistent enhancements across all models and subtasks. Compared with the other approaches, it again achieves the best performance on nearly every subtask, with most improvements exceeding 33.34%. These findings underscore the generalizability of our method and validate the effectiveness of categorizing video tasks by temporal variation characteristics, which enables more accurate guidance in selecting hallucination-sensitive attention heads for each sample.

Table 6: Results on VidHalluc benchmark with large VideoLLM.

Model	Variant	BQA	MCQ	STH	TSH	Overall
Qwen2.5-VL-72B	Baseline	81.17	75.65	66.75	76.00	74.89
	Ours	**84.25**	**86.16**	**69.43**	**78.52**	**79.59**

Scalability to Large VideoLLM. We further conducted additional experiments to verify the scalability of our method on the VidHalluc benchmark using the *Qwen2.5-VL-72B* model. The results are shown in Table 6. Firstly, it can be observed that while Qwen2.5-VL-72B outperforms smaller models like VideoLLaMA2-7B on most subtasks of the VidHalluc benchmark, indicating that more powerful models do indeed exhibit some mitigation of hallucination issues, hallucination problems still persist. Furthermore, for certain specific tasks (e.g., MCQ), the hallucination is even more severe compared to the VideoLLaMA2-7B. This highlights the persistent challenge of hallucination in VideoLLMs.

Secondly, the results demonstrate that our method also achieves significant performance improvements on the Qwen2.5-VL-72B model, demonstrating consistent hallucination mitigation capabilities across all subtasks. This proves the effectiveness and scalability of our method in large models.

Dataset statistics. We plot the accuracy of the binary classifier trained on $\mathcal{D}_t$ using VideoLLaMA2, as shown in Figure 10 on Appendix F. The accuracy curve follows the same trend as the accuracy curves of the STH (TSH) datasets $\mathcal{D}_S$ ($\mathcal{D}_T$), which validates the effectiveness of our data-collection pipeline. It can precisely gather samples with different temporal variation characteristics.

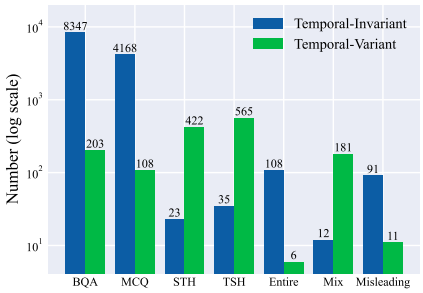


Figure 6: Temporal variation classifier predictions for each subtask.

Temporal Variation Classifier effectiveness. We next examine the performance of the trained classifier θ. Figure 6 reports its predictions on the subtasks of both benchmarks. In VidHalluc, BQA and MCQ are temporal-invariant tasks, and θ assigns more than 97% of their samples to the temporal-invariant category. In contrast, STH and TSH are temporal-variant tasks, and θ assigns more than 94% of their samples to the temporal-variant category. A similar pattern appears in EventHallusion: the Entire and Misleading focus on single events and thus align with temporal-invariant tasks, whereas the Mix involves multiple events and therefore aligns with temporal-variant tasks. The classifier successfully captures these distinctions.

Table 7: Results on VideoMMMU benchmark.

Model	Variant	Perception	Comprehension	Adaptation	Overall
VideoLLaMA2	Baseline	59.83	46.12	32.25	46.07
	Ours	59.47	46.24	32.25	45.99
Qwen2.5-VL	Baseline	57.29	42.93	37.16	45.79
	Ours	57.46	42.57	36.98	45.67
Video-LLaVA	Baseline	39.82	31.67	30.18	33.89
	Ours	39.82	31.59	31.09	34.17

Preservation of Model Capabilities. We conducted further evaluation on the VideoMMMU benchmark, a multi-modal and multi-disciplinary video benchmark that evaluates VideoLLMs' knowledge acquisition capability from educational videos, and the results are shown in Table 7.The results show that the performance of our method on all subtasks of the VideoMMMU benchmark is comparable to the Baseline, demonstrating that our approach does not impair the model's capabilities. Additional analysis and experimental results can be found in Appendix F.

5 Conclusion

In this study, we conduct the first systematic investigation into activation engineering feasibility and underlying mechanisms for mitigating hallucinations in VideoLLMs. We first perform an experimental analysis to identify the key factors that determine the effectiveness of activation engineering. Guided by these insights, we develop a temporal-aware activation engineering framework that dynamically identifies and manipulates hallucination-sensitive modules based on the temporal variation characteristics of task, without requiring additional LLM fine-tuning. Experimental results show that our method delivers significant performance improvements across all tasks, confirming its robustness in alleviating hallucination in VideoLLMs. Limitations can be found in Appendix G.

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Contract 623B2097, the Youth Innovation Promotion Association CAS. It was also supported by Merchants Union Consumer Finance Company Limited, and the GPU cluster built by MCC Lab of USTC & the Supercomputing Center of USTC.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

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

[4] Kyungho Bae, Jinhyung Kim, Sihaeng Lee, Soonyoung Lee, Gunhee Lee, and Jinwoo Choi. Mash-vlm: Mitigating action-scene hallucination in video-llms through disentangled spatial-temporal representations. *arXiv preprint arXiv:2503.15871*, 2025.

[5] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023.

[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. *arXiv preprint arXiv:2502.13923*, 2025.

[7] Zechen Bai, Pichao Wang, Tianjun Xiao, Tong He, Zongbo Han, Zheng Zhang, and Mike Zheng Shou. Hallucination of multimodal large language models: A survey, 2025.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[9] Zouying Cao, Yifei Yang, and Hai Zhao. Nothing in excess: Mitigating the exaggerated safety for llms via safety-conscious activation steering. *arXiv preprint arXiv:2408.11491*, 2024.

[10] Yue Chang, Liqiang Jing, Xiaopeng Zhang, and Yue Zhang. A unified hallucination mitigation framework for large vision-language models. *Transactions on Machine Learning Research*, 2024.

[11] Beitao Chen, Xinyu Lyu, Lianli Gao, Heng Tao Shen, and Jingkuan Song. Alleviating hallucinations in large vision-language models through hallucination-induced optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[12] Junzhe Chen, Tianshu Zhang, Shiyu Huang, Yuwei Niu, Linfeng Zhang, Lijie Wen, and Xuming Hu. Ict: Image-object cross-level trusted intervention for mitigating object hallucination in large vision-language models. *arXiv preprint arXiv:2411.15268*, 2024.

[13] Lin Chen, Xilin Wei, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Zhenyu Tang, Li Yuan, et al. Sharegpt4video: Improving video understanding and generation with better captions. *Advances in Neural Information Processing Systems*, 37:19472–19495, 2024.

[14] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.

[15] Xiang Chen, Chenxi Wang, Yida Xue, Ningyu Zhang, Xiaoyan Yang, Qiang Li, Yue Shen, Lei Liang, Jinjie Gu, and Huajun Chen. Unified hallucination detection for multimodal large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3235–3252, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

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

[17] Wey Yeh Choong, Yangyang Guo, and Mohan Kankanhalli. Vidhal: Benchmarking temporal hallucinations in vision llms. *arXiv preprint arXiv:2411.16771*, 2024.

[18] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[19] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[20] Ailin Deng, Zhirui Chen, and Bryan Hooi. Seeing is believing: Mitigating hallucination in large vision-language models via clip-guided decoding. *arXiv preprint arXiv:2402.15300*, 2024.

[21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[22] Jinhao Duan, Fei Kong, Hao Cheng, James Diffenderfer, Bhavya Kailkhura, Lichao Sun, Xiaofeng Zhu, Xiaoshuang Shi, and Kaidi Xu. Truthprint: Mitigating lvlm object hallucination via latent truthful-guided pre-intervention. *arXiv preprint arXiv:2503.10602*, 2025.

[23] Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[24] Alessandro Favero, Luca Zancato, Matthew Trager, Siddharth Choudhary, Pramuditha Perera, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. Multi-modal hallucination control by visual information grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14303–14312, June 2024.

[25] Jiztom Kavalakkatt Francis and Matthew J Darr. Interpretable ai for time-series: Multi-model heatmap fusion with global attention and nlp-generated explanations. *arXiv preprint arXiv:2507.00234*, 2025.

[26] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120, 2023.

[27] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[28] Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. Hallusionbench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14375–14385, June 2024.

[29] Anisha Gunjal, Jihan Yin, and Erhan Bas. Detecting and preventing hallucinations in large vision language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18135–18143, 2024.

[30] Evan Hernandez, Belinda Z Li, and Jacob Andreas. Inspecting and editing knowledge representations in language models. In *First Conference on Language Modeling*.

[31] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. Cogvlm2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*, 2024.

[32] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[33] Jitesh Jain, Jianwei Yang, and Humphrey Shi. Vcoder: Versatile vision encoders for multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27992–28002, 2024.

[34] Chaoya Jiang, Haiyang Xu, Mengfan Dong, Jiaxing Chen, Wei Ye, Ming Yan, Qinghao Ye, Ji Zhang, Fei Huang, and Shikun Zhang. Hallucination augmented contrastive learning for multimodal large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27036–27046, 2024.

[35] Prannay Kaul, Zhizhong Li, Hao Yang, Yonatan Dukler, Ashwin Swaminathan, CJ Taylor, and Stefano Soatto. Throne: An object-based hallucination benchmark for the free-form generations of large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27228–27238, 2024.

[36] Kai Konen, Sophie Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik Opitz, and Tobias Hecking. Style vectors for steering generative large language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 782–802, 2024.

[37] Ming Kong, Xianzhou Zeng, Luyuan Chen, Yadong Li, Bo Yan, and Qiang Zhu. Mhbench: Demystifying motion hallucination in videollms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 4401–4409, 2025.

[38] Seongyun Lee, Sue Hyun Park, Yongrae Jo, and Minjoon Seo. Volcano: Mitigating multimodal hallucination through self-feedback guided revision. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 391–404, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[39] Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13872–13882, 2024.

[40] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. Otter: A multi-modal model with in-context instruction tuning, 2023.

[41] Chaoyu Li, Eun Woo Im, and Pooyan Fazli. Vidhalluc: Evaluating temporal hallucinations in multimodal large language models for video understanding. *arXiv preprint arXiv:2412.03735*, 2024.

[42] 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 *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[43] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

[44] Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *ICLR*, 2023.

[45] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530, 2023.

[46] Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312, Toronto, Canada, July 2023. Association for Computational Linguistics.

[47] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[48] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[49] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26296–26306, June 2024.

[50] Sheng Liu, Haotian Ye, Lei Xing, and James Y Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *International Conference on Machine Learning*, pages 32287–32307. PMLR, 2024.

[51] Sheng Liu, Haotian Ye, and James Zou. Reducing hallucinations in large vision-language models via latent space steering. In *The Thirteenth International Conference on Learning Representations*, 2025.

[52] Fan Ma, Xiaojie Jin, Heng Wang, Yuchen Xian, Jiashi Feng, and Yi Yang. Vista-llama: Reducing hallucination in video language models via equal distance to visual tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13151–13160, 2024.

[53] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-ChatGPT: Towards detailed video understanding via large vision and language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12585–12602, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[54] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. Featured Certification.

[55] Yein Park, Chanwoong Yoon, Jungwoo Park, Minbyul Jeong, and Jaewoo Kang. Does time have its place? temporal heads: Where language models recall time-specific information. *arXiv preprint arXiv:2502.14258*, 2025.

[56] Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo Maria Ponti, and Shay Cohen. Spectral editing of activations for large language model alignment. *Advances in Neural Information Processing Systems*, 37:56958–56987, 2024.

[57] Ruchit Rawal, Reza Shirkavand, Heng Huang, Gowthami Somepalli, and Tom Goldstein. Argus: Hallucination and omission evaluation in video-llms. *arXiv preprint arXiv:2506.07371*, 2025.

[58] Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, 2024.

[59] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4035–4045, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[60] Pranab Sahoo, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11709–11724, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[61] Weichao Shen, Yuwei Wu, and Yunde Jia. Temporal invariant factor disentangled model for representation learning. In *Pattern Recognition and Computer Vision: Second Chinese Conference, PRCV 2019, Xi'an, China, November 8–11, 2019, Proceedings, Part II 2*, pages 391–402. Springer, 2019.

[62] Shashwat Singh, Shauli Ravfogel, Jonathan Herzig, Roee Aharoni, Ryan Cotterell, and Ponnurangam Kumaraguru. Representation surgery: Theory and practice of affine steering. In *Forty-first International Conference on Machine Learning*, 2024.

[63] Yiwei Sun, Zhihang Liu, Chuanbin Liu, Bowei Pu, Zhihan Zhang, and Hongtao Xie. Hallucination mitigation prompts long-term video understanding. *arXiv preprint arXiv:2406.11333*, 2024.

[64] Peter Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Adithya Jairam Vedagiri IYER, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

[65] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv e-prints*, pages arXiv–2308, 2023.

[66] Nasib Ullah and Partha Pratim Mohanta. Thinking hallucination for video captioning. In *Proceedings of the Asian Conference on Computer Vision*, pages 3654–3671, 2022.

[67] David Wan, Jaemin Cho, Elias Stengel-Eskin, and Mohit Bansal. Contrastive region guidance: Improving grounding in vision-language models without training. In *European Conference on Computer Vision*, pages 198–215. Springer, 2024.

[68] Xintong Wang, Jingheng Pan, Liang Ding, and Chris Biemann. Mitigating hallucinations in large vision-language models with instruction contrastive decoding. In *ACL (Findings)*, 2024.

[69] Yuxuan Wang, Yueqian Wang, Dongyan Zhao, Cihang Xie, and Zilong Zheng. Videohallucer: Evaluating intrinsic and extrinsic hallucinations in large video-language models. *arXiv preprint arXiv:2406.16338*, 2024.

[70] Jiarui Wu, Zhuo Liu, and Hangfeng He. Mitigating hallucinations in multimodal spatial relations through constraint-aware prompting. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3450–3468, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[71] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. VideoCLIP: Contrastive pre-training for zero-shot video-text understanding. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6787–6800, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[72] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[73] Dongjie Yang, Suyuan Huang, Chengqiang Lu, Xiaodong Han, Haoxin Zhang, Yan Gao, Yao Hu, and Hai Zhao. Vript: A video is worth thousands of words. *Advances in Neural Information Processing Systems*, 37:57240–57261, 2024.

[74] Tianyun Yang, Ziniu Li, Juan Cao, and Chang Xu. Mitigating hallucination in large vision-language models via modular attribution and intervention. In *The Thirteenth International Conference on Learning Representations*, 2025.

[75] Zhihe Yang, Xufang Luo, Dongqi Han, Yunjian Xu, and Dongsheng Li. Mitigating hallucinations in large vision-language models via dpo: On-policy data hold the key. *arXiv preprint arXiv:2501.09695*, 2025.

[76] Hongbin Ye, Tong Liu, Aijia Zhang, Wei Hua, and Weiqiang Jia. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794*, 2023.

[77] Shukang Yin, Chaoyou Fu, Sirui Zhao, Tong Xu, Hao Wang, Dianbo Sui, Yunhang Shen, Ke Li, Xing Sun, and Enhong Chen. Woodpecker: Hallucination correction for multimodal large language models. *Science China Information Sciences*, 67(12):220105, 2024.

[78] Qifan Yu, Juncheng Li, Longhui Wei, Liang Pang, Wentao Ye, Bosheng Qin, Siliang Tang, Qi Tian, and Yueting Zhuang. Hallucidoctor: Mitigating hallucinatory toxicity in visual instruction data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12944–12953, 2024.

[79] Zihao Yue, Liang Zhang, and Qin Jin. Less is more: Mitigating multimodal hallucination from an EOS decision perspective. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11766–11781, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[80] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025.

[81] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. In Yansong Feng and Els Lefever, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 543–553, Singapore, December 2023. Association for Computational Linguistics.

[82] Jiacheng Zhang, Yang Jiao, Shaoxiang Chen, Na Zhao, Zhiyu Tan, Hao Li, and Jingjing Chen. Eventhallusion: Diagnosing event hallucinations in video llms. *arXiv preprint arXiv:2409.16597*, 2024.

[83] Zhiyuan Zhao, Bin Wang, Linke Ouyang, Xiaoyi Dong, Jiaqi Wang, and Conghui He. Beyond hallucinations: Enhancing lvlms through hallucination-aware direct preference optimization. *arXiv preprint arXiv:2311.16839*, 2023.

[84] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[85] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

[86] Will Y Zou, Andrew Y Ng, and Kai Yu. Unsupervised learning of visual invariance with temporal coherence. In *NIPS 2011 workshop on deep learning and unsupervised feature learning*, volume 3, 2011.

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: We describe all claims in Abstract and Section 1.

   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: We provide a discussion on the limitations in Section G.

   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: [NA]

   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 fully disclose all information necessary to reproduce the main experimental results and validate the key conclusions of this paper, including the complete datasets and detailed hyperparameters described in Section 4 and Appendix E. The source code and datasets will be released upon acceptance.

   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: [Yes]

   Justification: We provide the complete settings used for all experiments in Section 4.1 and Appendix D.1 to reproduce all our experiments with instructions. All hyperparameters are described in Appendix E. The source code and datasets will be released upon acceptance.

   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: We provide the complete settings used for all experiments in Section 4.1 and Appendix D.1 to reproduce all our experiments with instructions. All hyperparameters are described in Appendix E. The source code and datasets will be released upon acceptance.

   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: [No]

Justification: We conduct experiments across multiple models and benchmarks to validate the effectiveness of our method. Due to constraints in time and computational resources, error bars are not included.

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: We provide an estimate of the required computational resources in Appendix E.

   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: We have read the NeurIPS Code of Ethics.

   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 discuss both potential positive societal impacts and negative societal impacts in Appendix H.

    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: We focuses on mitigating hallucinations in existing open-source VideoLLMs using activation engineering during inference, without altering model generation behavior or releasing new models or datasets. Therefore, all existing safeguards (if any) remain in place.

    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 explicitly cite dataset in Section 4.1. And all models and benchmarks are listed and cited in Section 4.2.

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: [Yes]

Justification: We will release our code and datasets for easy use via Github and Hugging Face.

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: We don't 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: We don't 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: We provide detailed descriptions of the usage of existing LLMs in Section 4 and Appendix D.

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.

A Related Work

A.1 MLLMs for Video Understanding

Following the remarkable success of Large Language Models (LLMs) in natural language processing [1, 27, 8, 18, 26, 48, 72, 3], researchers have explored the integration of visual information into LLMs to endow them with broader capabilities, giving rise to the concept of multimodal LLMs (MLLMs) [19, 53, 2, 32, 14, 64, 43, 84]. In the image domain, a common approach is to divide an image into patches, treat each patch as a token [21], and employ a visual encoder followed by a non-linear alignment module to align the visual representation with the LLM's token space [31, 40, 5, 49]. More recently, this paradigm has been extended to the video domain, referred to as VideoLLM [53, 47, 6, 80]. Unlike images, videos introduce a temporal dimension, prompting the development of various methods for handling temporal information. For instance, Video-LLaVA [47] employs a time-sensitive attention mechanism to model frame-wise variations; the VideoLLaMA series [81, 16, 80] leverages Q-formers [42] or convolutional downsampling to model temporal features while reducing the number of generated tokens, thereby mitigating computational overhead; and Qwen2.5-VL [6] utilizes dynamic frame rate training along with absolute time encoding to support native dynamic resolution and frame rate adaptation.

A.2 Hallucinations in MLLMs

Despite their impressive achievements, MLLMs are prone to severe hallucination issues [60, 78, 7, 76], where the model fails to correctly interpret input data and generates inaccurate or entirely fabricated responses. To mitigate hallucinations in the image domain, a variety of approaches have been proposed, which can generally be categorized into two types: additional training-based methods [34, 75, 35, 24] and training-free methods [74, 10, 38, 77, 67]. The first category focuses on constructing supplementary training data to further align model outputs, albeit at the cost of significant computational resources. The second category aims to reduce hallucinations during inference without requiring model retraining. For example, Volcano [38] and Woodpecker [77] adopt self-feedback mechanisms to iteratively refine model responses, while VCD [39] and ICD [68] employ contrastive decoding [46] to suppress the probability of hallucinated tokens. Other approaches include hallucination detection and correction [15, 29], as well as enhancing the model's attention to image content using auxiliary signals such as attention weights [83, 33, 11].

When extending MLLMs to the video domain, hallucination becomes even more complex due to the inherent temporal dynamics of video data [52, 82, 17, 69]. To address this, researchers have adapted and extended image-based techniques, including collecting high-quality datasets for fine-tuning [4, 66, 52, 38], applying self-feedback mechanisms [79], leveraging contrastive decoding strategies [82], detecting and correcting hallucinations [22], and utilizing auxiliary information to strengthen the model's understanding of video content [41]. For instance, Vript [73] retrained the model on a $12k$ example high-quality dataset to diminish hallucination behaviour. VISTA-LLAMA [52] removed relative position encodings between visual and textual tokens and performs further training so that visual tokens exhibit a stronger influence during text generation. Zihao et al. [79] insert a special EOS token that encourages the model to terminate its output and self-assess the completeness of the generated content. TCD [82] adopts contrastive decoding to decrease the probability of producing hallucinated tokens. DINO-HEAL [41] re-weights video embeddings with attention maps generated by an auxiliary DINO [54] model, thereby attenuating the impact of irrelevant information. In parallel, a range of benchmarks has been introduced to evaluate hallucination in VideoLLMs [41, 82, 69, 17, 73]. VidHal [17] focuses on temporal hallucination, EventHallusion [82] targets hallucination related to events, and VidHalluc [41] provides a comprehensive evaluation of hallucinations in motion details and spatiotemporal consistency. Nevertheless, compared to the image domain, research on hallucination in VideoLLMs remains relatively underexplored.

A.3 Activation Engineering

Activation engineering [65, 85, 44, 30, 58] refers to manipulating the activations within a model to control its behavior. Prior works have demonstrated its effectiveness on truthfulness [45], formality transfer [50], sentiment control [65, 36] and refusal behavior control [9].

ITI [45] was the first to apply this technique to mitigate textual hallucinations in LLMs by computing activation vectors from normal samples and hallucination-inducing samples, and injecting them into the multi-head attention layers during inference to suppress hallucinated content. More recently, several studies have extended activation engineering to address hallucinations in MLLMs for image-related tasks [51, 12]. Specifically, ICT [12] computes image-level and object-level vectors, and leverages the ITI method to reduce both global and local hallucinations. VTI [51] builds upon this by injecting activation vectors into both the vision encoder layers and the LLM layers, thereby mitigating hallucinations more effectively.

However, the feasibility of applying activation engineering to VideoLLMs remains an open question. In this paper, we explore the applicability of activation engineering for mitigating hallucinations in VideoLLMs. In addition, we examine the key factors that influence its effectiveness.

A.4 Activation Editing Methods

Recent works have explored activation editing techniques from spectral, geometric, and constrained perspectives. Spectral Editing of Activations (SEA) [56] leverages SVD-based inference-time projections on cross-covariance matrices, achieving data-efficient edits without training. AlphaEdit [23] introduces null-space constrained weight updates, providing theoretical guarantees against catastrophic forgetting during sequential edits. Singh et al. [62] formulates steering transformations as optimal affine mappings over hidden states, effectively addressing biases and toxic outputs. Unlike these approaches, our method dynamically identifies hallucination-sensitive modules in VideoLLMs via temporal-aware activation editing, adapting inference-time edits specifically to temporal variation characteristics in video tasks.

B Basic Process of Activation Engineering

Activation engineering requires the construction of a curated dataset $\mathcal{D} = \{(x_n^{(i)}, y^{(i)}), (x_h^{(i)}, y^{(i)})\}$, $i = \{1, 2, ..., |\mathcal{D}|\}$, where $x_n^{(i)}$ denotes a normal prompt, $x_h^{(i)}$ denotes a hallucination-inducing prompt, and $y^{(i)}$ is the corresponding ground-truth response. For each data pair, both $(x_n^{(i)}, y^{(i)})$ and $(x_h^{(i)}, y^{(i)})$ are passed through the LLM, and the model's internal vectors from N distinct modules at the position of the final token are extracted, denoted as $\mathcal{V}_n^{(j)} = \{v_n^{(i,j)}\}$ and $\mathcal{V}_h^{(j)} = \{v_h^{(i,j)}\}$, where $j = \{1, 2, ..., N\}$. This process yields a set of vectors for each module: $\mathcal{V}_{nh}^{(j)} = \{v_n^{(i,j)}, v_h^{(i,j)}\}$. Since $v_n^{(i,j)}$ and $v_h^{(i,j)}$ represent the normal latent states and hallucination-prone latent states respectively, an offset vector $v^{(i,j)} = v_n^{(i,j)} - v_h^{(i,j)}$ is computed to characterize the transition from the hallucinated latent space to the normal latent space. Aggregating these across the dataset yields offset vector sets $\mathcal{V}^{(j)} = \{v^{(i,j)}\}$, and an average offset vector $v^{(j)} = \sum_{i=1}^{|\mathcal{D}|} v^{(i,j)}/|\mathcal{D}|$ is computed per module to obtain the final offset vector set $\mathcal{V} = \{v^{(j)}\}$, with one vector per module. During inference, these offset vectors $v^{(j)}$ are injected into their corresponding modules within the LLM to suppress hallucination behaviors.

As illustrated in Figure 1, activation engineering can be categorized into Attention Head Engineering and Layer Engineering, depending on the position of the modules selected for injection. The former targets individual attention heads, while the latter operates at the level of transformer layers. Taking Attention Head Engineering as an example, for an LLM with L transformer layers and M attention heads per layer, a total of $N = L \times M$ offset vectors need to be computed. Prior studies have shown that different attention heads exhibit varying levels of sensitivity to hallucinations. Therefore, instead of applying offset vectors to all attention heads, we selectively compute offset vectors only for those with strong hallucination sensitivity, thereby reducing unnecessary computation.

Specifically, for each attention head j, we utilize its associated vector set $\mathcal{V}_{nh}^{(j)} = \{v_n^{(i,j)}, v_h^{(i,j)}\}$ to train a binary classifier that determines whether the head is sensitive to hallucinations. We then select the top-K attention heads with the highest classification accuracy and apply offset vector injection only to these heads during VideoLLM inference.

C Intuitive Explanation of Hallucination Sensitivity and Offset Injection

Task-agnostic but temporally correlated hallucination sensitivity. Firstly, one of the core challenges for VideoLLMs is effectively capturing and integrating the temporal dynamics of videos. When temporal information is incomplete or misinterpreted, the model is more prone to generating hallucinations [37, 57]. Conversely, task types often represent a higher-level semantic abstraction that is largely independent of the underlying video's intrinsic temporal characteristics. For a given video, the same underlying temporal events can be probed with different task formats. For example, a multiple-choice question can be easily rephrased into a true/false question without altering the video's temporal content.

The effectiveness of injecting offset vectors. Secondly, regarding the effectiveness of injecting offset vectors, we can view the attention heads and layers within a VideoLLM as "experts" that process information at different levels of abstraction. Some of these "experts" may be specifically responsible for handling temporal dynamics or integrating information across time, which is consistent with previous findings [25, 55]. When these "experts" receive ambiguous or inconsistent temporal signals (e.g., due to frame downsampling), they may produce biased internal representations, leading to hallucinations. By calculating and injecting "offset vectors", we are essentially introducing a "correction signal" into the internal workspace of these critical "experts". This signal pushes the activations towards a state closer to the true, non-hallucinated output, thereby correcting the "latent deviation" caused by incomplete or misleading temporal information. This has also been demonstrated in previous works [12, 45]. This targeted intervention, combined with our discovery of the modules' sensitivity to temporal variations, enables activation engineering to efficiently and specifically mitigate hallucination problems in VideoLLMs.

Temporal Variation Classification Prompt

```
This video comes with a description that summarizes its content. Please analyze whether there are
any
**significant scene changes** based on both the description and the extracted frames.

A **scene change** is defined as one of the following:
1. **Event change**: Different activities or events occur across frames, such as switching to a
new place,
time, or a different plot segment.
2. **Viewpoint or camera change**: The same event is shown, but from a noticeably different
perspective
(e.g., switching from stage view to audience view, or from wide shot to close-up).

Here is the description:
"{description}"

Please answer in **strict JSON format**, with the following fields:\n
```
{{
    \"scene_change\": true or false,
    \"reason\": \"A short explanation of your reasoning.\"
}}
```
Return only valid JSON. Do not include any other commentary or explanation.
{video}
```

Figure 7: Temporal variation classification prompt for identifying the temporal variation characteristics of each sample.

D Additional Details of Temporal-Aware Activation Engineering Framework

D.1 Data Cleaning

Although ShareGPT4Video originally contains $40k$ (video, question, answer) triples, activation engineering benefits more from a compact, high-quality corpus. We begin with dataset cleaning through a multi-step process. First, duplicate videos are identified and removed to ensure uniqueness. Next, we filter out clips with low visual quality (e.g., excessive blur, compression artifacts) or incomplete content that may hinder understanding. For the remaining videos, we inspect associated metadata, such

as titles, descriptions, and timestamps, and discard entries with missing or inconsistent information. Finally, we apply automated heuristics to detect and eliminate residual noisy or anomalous samples, such as videos with corrupted frames or mismatched audio-visual alignment. These operations reduce the collection to $10k$ items, which we denote as $\mathcal{D}_1$.

D.2 Filtering Datasets for Distinct Task Types

Figure 7 illustrates the temporal variation classification prompt we used to judge whether a task is temporal-invariant or temporal-variant.

D.3 Dataset Statistics

In our experiments, τ is set to 0.8. The dataset $\mathcal{D}_a^f$ contains 422 sample instances, while the dataset $\mathcal{D}_t^f$ contains 574 sample instances. The average video length varies between the two datasets: $\mathcal{D}_a^f$ videos have an average length of 104.37 frames, whereas $\mathcal{D}_t^f$ videos have an average length of 786.49 frames. These statistics highlight the rationality of our constructed datasets, where $\mathcal{D}_a^f$ videos are shorter and exhibit temporal-invariant characteristic, while $\mathcal{D}_t^f$ videos are longer and demonstrate temporal-variant characteristic.

E Detailed Experimental Settings

Normal and hallucination-inducing prompt. Consistent with Section 3.2, we use concatenating the original video and the question as the normal prompt x, and use concatenating the $4x$ downsampled video and the question as the hallucination-inducing prompt x_h.

Extending CGD and VTI to the video domain. Since both CGD and VTI are primarily image-based methods, we adapted them to the video domain with the following improvements:

- For CGD, we replaced the CLIP model used for similarity score calculation with VideoCLIP model [71].
- For VTI, we masked all frames of each video to generate a set of corresponding random masked videos. Additionally, for each video caption x, we used GPT-4o [32] to generate its corresponding hallucinated version $\tilde{x}$.
- For a fair comparison, we used the union of $\mathcal{D}_a^f$ and $\mathcal{D}_t^f$ to generate VTI's visual and textual directions.
- All other hyperparameters adopted the optimal values provided in their respective papers.

Hyperparameter tuning. In our experiments, we employ greedy decoding to ensure reproducibility and perform grid search for hyperparameter tuning, systematically exploring combinations of key parameters. The main hyperparameters include K, the number of top-ranked attention heads selected, and α, the scaling factor applied to the offset vectors. The search space is defined as the Cartesian product:

$$\{32, 64, 128, 256\} \times \{8, 16, 24, 32\}, \tag{1}$$

where resulting in $4 \times 4 = 16$ unique configurations, a relatively small search space. For compared methods, we also apply grid search for fair comparison. For TCD, we tune the frame downsampling rate r, as well as the contrastive decoding parameters α and β, over the space:

$$\{2, 4, 8\} \times \{0.25, 0.5, 0.75, 1.0\} \times \{0.1, 0.5\}. \tag{2}$$

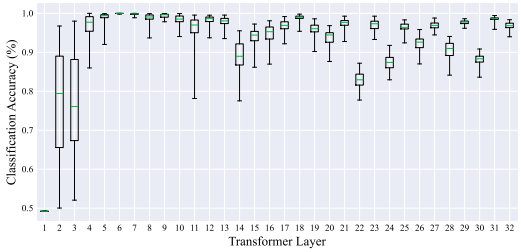
It yielding $3 \times 4 \times 2 = 24$ configurations. For DINO-HEAL, we search over combinations of normalization (enabled or not) and three DINO variants with or without registers, leading to $2 \times 2 \times 3 = 12$ configurations in total.

Temporal Variation Classifier. Owing to VideoLLaMA2's strong spatiotemporal representation and low inference latency, we utilize its submodules to construct θ. Specifically, we select VideoLLaMA2's visual encoder and the first transformer layer of its LLM as the backbone for θ, followed

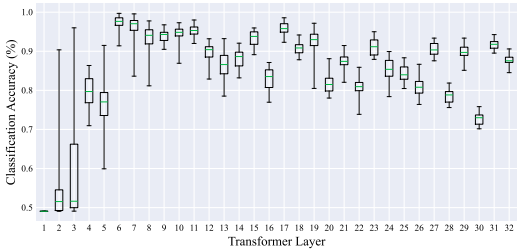
by a linear classification layer that performs classification based on the output of the last token. For classifier training, to reduce computational overhead, we freeze the backbone of the classifier and only train its final classification layer. The learning rate is set to 1×10^{-5}, followed by a cosine learning rate schedule with an initial warmup of 10 steps and a batch size of 8. The classifier is trained for 5 epochs.

All experiments are conducted on a single machine with 8 NVIDIA A100 80 GB GPUs.

F Additional Experimental Results

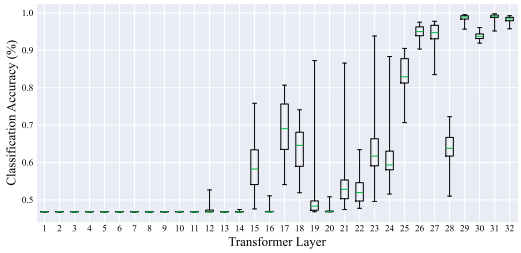


(a) Layer-wise distribution of binary-classifier accuracy across attention heads trained on $\mathcal{V}_{nh,B}$.

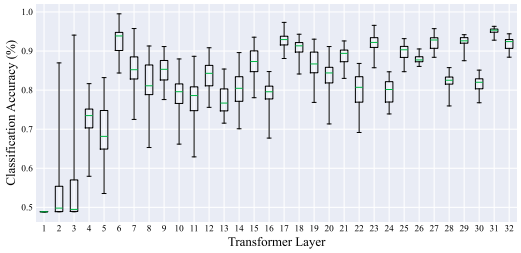


(b) Layer-wise distribution of binary-classifier accuracy across attention heads trained on $\mathcal{V}_{nh,M}$.

Figure 8: Binary classifier performance trained on vector sets $\mathcal{V}_{nh,B}$ and $\mathcal{V}_{nh,M}$.



(a) Layer-wise distribution of binary-classifier accuracy across attention heads trained on $\mathcal{V}_{nh,T}$.



(b) Layer-wise distribution of binary-classifier accuracy across attention heads trained on $\mathcal{V}_{nh}$.

Figure 9: Binary classifier performance trained on vector sets $\mathcal{V}_{nh,T}$ and $\mathcal{V}_{nh}$.

Table 8: Ablation results on hyperparameter sensitivity in the VidHalluc benchmark.

α Range/Top-K Attention Head Selection Number	32	64	128	256
8	71.25	71.56	71.94	71.47
16	71.28	71.79	72.08	71.63
24	71.72	71.98	72.17	71.84
32	71.74	72.03	72.49	71.96

Ablation on hyperparameter sensitivity. We further conducted an ablation study on the value range of the injection weight (α) and the number of Top-K attention heads selected. Specifically, following the grid search settings in the paper, we performed a grid search for the injection weight in the range $[8, 16, 24, 32]$ and for the Top-K attention head selection number in the range $[32, 64, 128, 256]$, using the VideoLLaMA2 model on the VidHalluc benchmark. The experimental results are shown in Table 8 (using *Overall* as the evaluation metric). As can be seen, while the injection weight (α) and the number of Top-K attention heads do influence model performance, the sensitivity is not drastic. Within a reasonable range, model performance generally improves as both the injection weight and the number of selected Top-K attention heads increase. However, when more attention heads (256) that are less sensitive to hallucinations are included, there is a slight decrease in performance. In practical applications, an accuracy threshold strategy can also be adopted to select attention heads, further enhancing the robustness of the model.

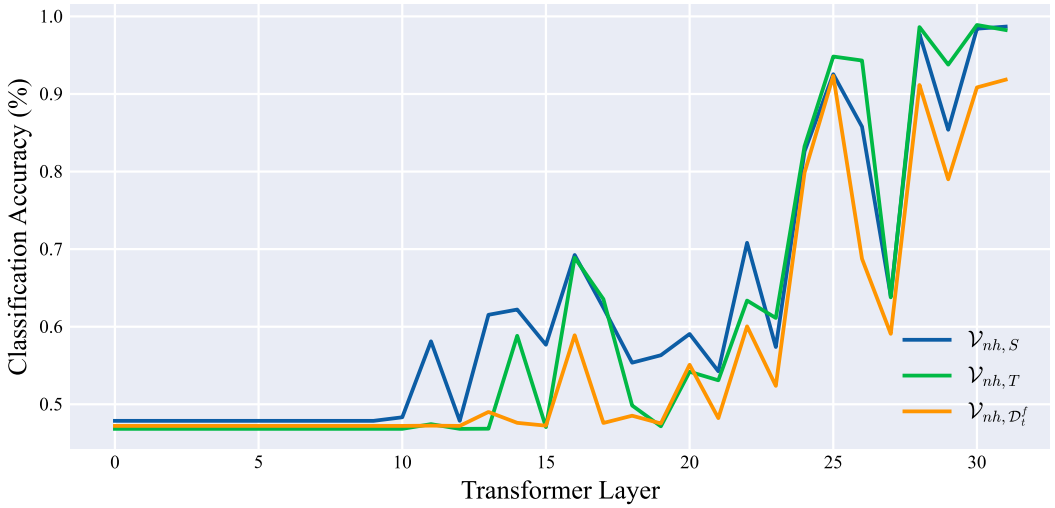


Figure 10: Layer-wise mean binary-classifier accuracy across attention heads trained on vector sets $\mathcal{V}_{nh,*}$ from $\mathcal{D}_S$, $\mathcal{D}_T$, and $\mathcal{D}_t^f$.

Table 9: Results on VidHalluc and EventHallusion benchmarks with VideoLLaMA2.

Method	VidHalluc					EventHallusion			
	BQA	MCQ	STH	TSH	Overall	Entire	Mix	Misleading	Overall
Ours	79.09	84.07	65.14	61.67	72.49	43.86	70.47	57.70	59.17
Ours (Oracle Classification)	78.46	83.98	64.88	60.50	72.00	42.98	67.87	54.90	57.70

Ablation without pre-classification. We conduct an additional study in which we skip the classifier and directly select the data source according to the above analysis in Section 4.3. Specifically, for BQA and MCQ in VidHalluc and Entire and Misleading in EventHallusion, we perform activation engineering using $\mathcal{D}_a$; for STH and TSH in VidHalluc and Mix in EventHallusion, we use $\mathcal{D}_t$, denoting this setting as *Ours (Oracle Classification)*. As shown in Table 9, the results of *Ours (Oracle Classification)* closely match those of *Ours*, further confirming both the quality of our dataset and the effectiveness of the classifier θ.

Table 10: Per-sample inference time on the VidHalluc benchmark (unit: seconds).

Model	Variant	BQA	MCQ	STH	TSH
VideoLLaMA2	Baseline	1.44	1.66	1.87	1.92
	Ours	1.44	1.66	1.86	1.93
Qwen2.5-VL	Baseline	0.99	1.00	1.35	1.10
	Ours	0.99	1.00	1.34	1.10
Video-LLaVA	Baseline	2.43	2.49	3.18	3.86
	Ours	2.43	2.49	3.19	3.88

Inference efficiency. We further evaluated the per-sample inference time of both our method and the Baseline on the VidHalluc benchmark. To ensure fairness, each model was tested on a single 3090 GPU, eliminating any additional overhead from distributed inference. As described in Section 4.1, our method reduces inference-time overhead by executing the classifier and visual encoder in parallel. The results, presented in Table 10, also show that the per-sample inference time of our method is comparable to that of the Baseline, confirming the efficiency of our parallel design.

Classifier sample efficiency. We further tested the classifier's sample efficiency by reducing the number of training samples. Specifically, we trained with 100, 200, 300, and 400 samples respectively, and recorded the classification accuracy of the classifier on the VidHalluc benchmark for each quantity.

Table 11: Sample efficiency results of the classifier on the VidHalluc benchmark.

Sample Size	BQA	MCQ	STH	TSH
100	90.17%	89.63%	88.09%	88.83%
200	92.13%	91.47%	91.01%	90.33%
300	94.26%	95.39%	92.58%	92.50%
400	97.63%	97.47%	94.83%	94.17%

The experimental results are shown in Table 11. It can be observed that as the number of samples increases, the classifier's accuracy gradually improves. However, even with only 100 training samples, the classifier's accuracy still exceeds 88%. This demonstrates that our classifier possesses high sample efficiency.

Table 12: Hallucination consistency in failure cases on VidHalluc benchmark.

Model	BQA	MCQ	STH	TSH	Overall
Qwen2.5-VL-7B	95.24%	92.37%	96.13%	95.92%	94.92%

Failure cases. We further analyzed the specific changes in model output when using the Baseline versus our method, across all subtasks' failure cases on the VidHalluc benchmark with the Qwen2.5-VL-7B. Specifically, we used GPT-4o to determine whether the hallucinations produced by the Baseline and our method's outputs were consistent, and we calculated their hallucination consistency rate. The experimental results are shown in Table 12. As can be seen, the hallucination consistency rate of our method with the Baseline exceeds 90% across all subtasks' failure cases. This indicates that in the vast majority of failure situations, our method gracefully degrades to the original model's behavior and does not further exacerbate the hallucinations.

Ground Truth: Scene change: Yes, Locations: from on a stage to in the woods.

Model Output (Baseline): Scene change: Yes, Locations: **from a dark room to a car**.

Model Output (Ours): Scene change: Yes, Locations: **from performing on a stage to being in the woods**.

Figure 11: Case study of hallucination reduction on the STH description task.

Case Study. We present a concrete example to illustrate how our method reduces hallucinations. Specifically, since most subtasks are selection/classification tasks, we will focus on providing specific examples from the STH's description task. We demonstrate the specific changes in model output after using the baseline and after applying our method to more intuitively understand how our approach mitigates hallucinations. The example is shown in Figure 11. As can be seen, the Baseline exhibited hallucinations regarding the order of scene changes and scene recognition, while our method effectively reduced these hallucinations.

G Limitations

Despite the substantial improvements achieved by our method, several limitations remain to be addressed. First, our current approach adopts a coarse-grained taxonomy that partitions video tasks into only two categories based on their temporal variation characteristics: temporal-invariant and temporal-variant. While this dichotomy effectively captures broad distinctions in hallucination sensitivity, it oversimplifies the inherent heterogeneity within and across tasks. We hypothesize that developing a more fine-grained taxonomy could further enhance the adaptability and efficacy of activation engineering in mitigating hallucinations.

Second, although we employ optimization strategies such as pre-computing offset vectors and overlapping the classifier's execution with the model's forward pass, our method still introduces slight computational overhead during inference. Addressing this limitation calls for future research into more efficient activation engineering paradigms.

H Societal Impacts

H.1 Positive Impacts

The proposed temporal-aware activation engineering framework significantly mitigates hallucinations in VideoLLMs without requiring additional model fine-tuning. This advancement enhances the reliability and factual accuracy of AI-generated video understanding, benefiting applications such as video-based education, content summarization, and human-computer interaction. By addressing hallucination issues, the work promotes safer deployment of AI in sensitive domains like healthcare, security surveillance, and autonomous systems, where erroneous outputs could have serious consequences. Additionally, the method's train-free nature reduces the environmental and financial costs typically associated with large-scale model retraining, contributing to the sustainable development of AI technologies.

H.2 Negative Impacts

Despite its advantages, the method introduces extra computational overhead during inference, which could limit its scalability in real-time applications or resource-constrained environments. Furthermore, improving the realism and believability of AI-generated video outputs might inadvertently facilitate malicious uses, such as generating more convincing deepfakes or disinformation. Finally, while the work mitigates hallucinations, it does not eliminate them entirely, and overreliance on such systems without proper human oversight could lead to unforeseen societal risks.