CONCEPTTUNE: TUNING CONCEPT VECTORS FOR GENERALIZABLE VIDEO UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite their impressive capabilities, multimodal large language models (MLLMs) still face challenges in video understanding, which current approaches mainly address through parameter fine-tuning, represented by LoRA and LOFIT, or activation steering, represented by RepE. The former enables accurate task-specific adjustment by updating model weights or parameter offsets, but may alter pretrained capabilities. The latter offers flexible generalization through direct representation-space intervention, but their positive–negative activation contrasts may not accurately represent the target concept. We propose **ConceptTune**, a hybrid method combining the accuracy of fine-tuning with the generalization ability of activation steering. ConceptTune fine-tunes a learnable concept offset directly in model activations rather than model parameters, treating this offset as the concept. Each concept offset is learned from many examples through concept-targeted generative completion. This training procedure encourages the concept offset to capture semantic information shared across training examples of the same concept type, yielding a reusable concept representation that can be applied as a lightweight, plug-and-play activation intervention at inference time. Experiments show that ConceptTune improves transfer across video-grounding benchmarks and compares favorably with parameter-efficient fine-tuning and contrastive activation steering, supporting concept-targeted learning in activation space.

1 INTRODUCTION

Multimodal large language models (MLLMs) have achieved strong performance on vision-language tasks by combining visual encoders with instruction-following language models (Liu et al., 2023; Li et al., 2023a; Dai et al., 2023). Recent video-capable MLLMs extend this progress to video understanding, yet their performance remains less reliable on tasks that require detailed understanding of visual content and temporal events. Temporal moment retrieval, highlight detection, and video question answering, for example, require models to recognize visual content, integrate evidence across frames, and express their predictions in different output formats, including temporal spans, highlight scores, and natural-language answers (Gao et al., 2017; Lei et al., 2021; Xiao et al., 2021; Liu et al., 2024). Improving video understanding while preserving the capabilities of the underlying MLLM remains an important challenge.

A common way to improve a pretrained MLLM is to fine-tune a compact set of parameters (Houlsby et al., 2019; Li & Liang, 2021; Lester et al., 2021). Parameter-efficient methods such as LoRA (Hu et al., 2022) and LOFIT (Yin et al., 2024) use gradient-based optimization to update low-rank weight components or localized representation offsets, enabling accurate task-specific adaptation. However, the learned updates are typically optimized for a particular adaptation task and output format and may affect pretrained capabilities. A different line of work performs activation intervention. Methods such as RepE (Zou et al., 2023) derive steering directions from positive–negative activation contrasts and inject them into intermediate representations at inference time, providing a lightweight and flexible intervention interface. However, the resulting direction depends on how the contrasting examples are constructed. In video, positive and negative examples may differ not only in the intended content but also in the actor, scene, camera motion, or temporal context, making it difficult for the resulting direction to isolate the intended semantics. These complementary strengths and limitations motivate combining gradient-based optimization with reusable activation intervention.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

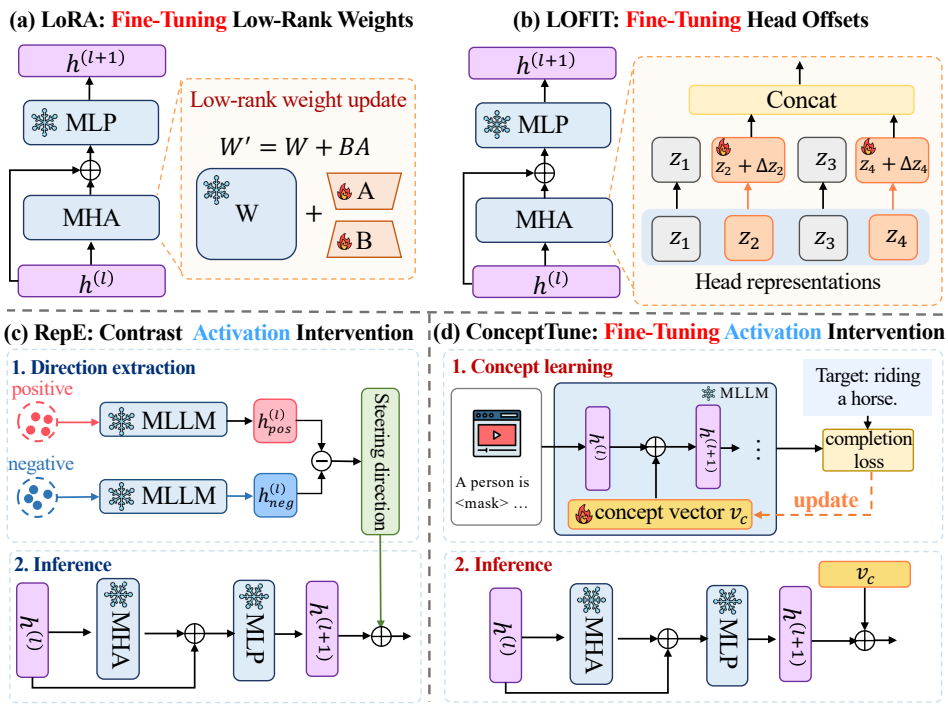


Figure 1: Comparison of lightweight adaptation interfaces. LoRA fine-tunes low-rank weight updates, LOFIT fine-tunes localized offsets at selected attention heads, and RepE derives a steering direction from contrasting activations. ConceptTune instead fine-tunes a shared concept vector directly in the residual-stream activations of a frozen video MLLM and reuses it as an activation intervention at inference time.

We introduce **ConceptTune**, which combines gradient-based concept learning with activation intervention. Rather than updating the MLLM parameters, ConceptTune fine-tunes a learnable offset added directly to residual-stream activations and treats this offset as a concept vector. For each concept type, the same layer-indexed vector is shared across many training examples and optimized through concept-targeted generative completion. We mask expressions of the corresponding concept type and train the frozen MLLM to reconstruct them; action and object phrases are masked directly, while temporal concepts are supervised through coarse temporal-position completion. Because the same vector must support the reconstruction of diverse expressions across examples, this training procedure encourages it to capture semantic information shared by the concept type rather than details specific to an individual video or description. An answer-only loss focuses learning on the designated targets without requiring positive–negative video pairs. Once learned, the concept vectors are applied as lightweight activation interventions and reused within the same backbone across datasets and task formats, without target-side training or tuning.

We evaluate ConceptTune on three video MLLMs and study the improvement and transfer of its learned concept vectors. Action and object concept vectors improve temporal moment retrieval and highlight detection across multiple benchmarks. Action and object vectors learned from grounding supervision also transfer to VideoQA under a different output format, without VQA-specific training. Moreover, independently learned concept vectors can be orthogonalized and composed to enhance multiple video concepts simultaneously. Comparisons with contrastive RepE and matched-supervision experiments with LoRA and LOFIT demonstrate the benefit of combining concept-targeted gradient-based learning with activation-level intervention. Our contributions are as follows:

1. **Activation-level concept learning.** We introduce ConceptTune, a lightweight framework that combines gradient-based learning with activation intervention by fine-tuning concept vectors in intermediate activations while keeping the video MLLM backbone frozen.

2. **Concept-targeted generative supervision.** We propose concept-targeted generative supervision, in which a concept vector is shared and optimized across examples by reconstructing expressions of the corresponding concept type. This enables concept learning without constructing positive–negative activation pairs and supports reuse across datasets and task formats.
3. **Cross-task transfer for video understanding.** We demonstrate transferable video understanding improvements across three MLLMs, including cross-task transfer to VideoQA, with comparisons against RepE, LoRA, and LOFIT.

2 Related Work

Parameter-efficient adaptation. Parameter-efficient fine-tuning methods reduce adaptation cost by restricting the trainable components of a pretrained model. LoRA (Hu et al., 2022) learns low-rank updates for selected weight matrices, while BitFit (Zaken et al., 2022) updates only bias parameters. LOFIT (Yin et al., 2024) localizes adaptation by learning offsets for selected attention-head representations. Representation Editing (RED) (Wu et al., 2024a) instead adapts frozen models by learning shared scaling and bias transformations on intermediate activations. ReFT (Wu et al., 2024b) similarly learns task-specific interventions on hidden representations. These methods retain most pretrained parameters and use gradient-based objectives to acquire task-relevant behavior. ConceptTune follows this parameter-efficient optimization principle while treating external concept vectors in the residual stream as the trainable objects. The vectors are learned through concept-targeted video completion and can be stored independently of the backbone for reuse across downstream tasks.

Representation editing and activation steering. A complementary line of work controls model behavior by modifying intermediate representations rather than updating model weights. Representation engineering methods such as RepE (Zou et al., 2023) and Contrastive Activation Addition (Panickssery et al., 2023) derive steering directions from activation contrasts and inject them during inference. Related studies have used inference-time activation intervention to improve truthfulness in language models (Li et al., 2023b), adapt vision-language representations (Bi et al., 2025), mitigate hallucinations (Su et al., 2025), and control multimodal model behavior (Sivakumar et al., 2025). These methods demonstrate that intermediate activations provide a lightweight control interface. ConceptTune adopts this interface but learns its directions through concept-targeted generative completion, using the target continuation itself as the optimization signal rather than constructing positive–negative activation pairs.

Video concept supervision. Video grounding benchmarks study whether models can align language queries with visual content over time. Charades-STA and QVHighlights evaluate temporal moment retrieval and highlight detection, respectively (Gao et al., 2017; Lei et al., 2021). Recent video-capable MLLMs address these tasks with task-specific training and structured temporal prediction (Guo et al., 2025; Huang et al., 2024; Qian et al., 2024; Wang et al., 2024). Their supervision is usually tied to a particular output format, such as temporal spans, highlight scores, or structured answers. ConceptTune instead uses grounding annotations to construct completion targets for reusable video concepts, including actions, objects, and coarse temporal positions. The resulting vectors are applied through the same activation interface across datasets and task formats.

3 ConceptTune

ConceptTune learns activation-level concept vectors through the generative interface of a frozen video MLLM and applies them across downstream tasks. Figure 2 summarizes the concept supervision, concept-vector learning, and target-time reuse stages.

3.1 Learning Video Concept Vectors

Let f_θ denote a frozen video MLLM with L Transformer layers and hidden width d. Given a video V and a textual prompt Q, let $H^{(l)} \in \mathbb{R}^{T\times d}$ denote the residual-stream representation at layer l, where T is the number of token positions at that layer. For each video concept type $c \in \mathcal{C}$ and

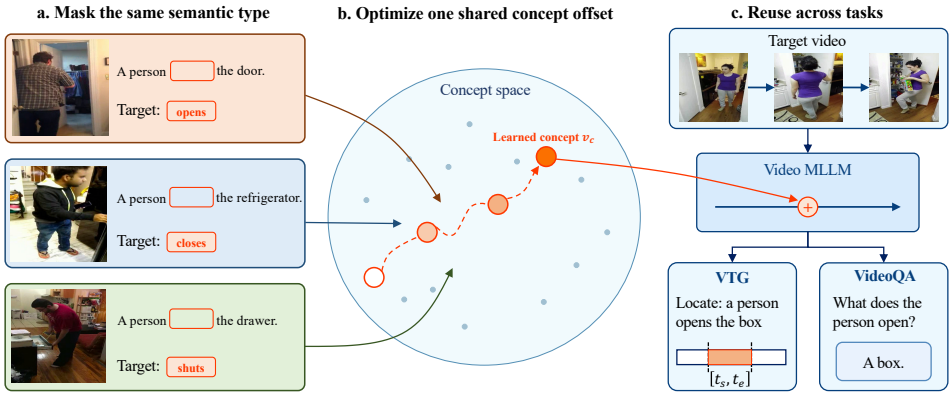


Figure 2: Illustration of ConceptTune. (a) Concept-targeted completion illustrated with action examples. (b) Diverse video–prompt pairs supervise a shared concept vector through instance-specific completions. (c) The learned vector is reused for temporal grounding and VideoQA without target-side tuning. The activation space is shown schematically.

candidate layer $l \in \mathcal{K}$, ConceptTune introduces an external concept vector $v_c^{(l)} \in \mathbb{R}^d$. At each candidate layer, the same vector is shared across all training examples associated with concept type c; it is not generated separately for each video or target expression.

During concept learning, an intervention hook adds the corresponding vector to the residual-stream representation at every candidate layer in $\mathcal{K}$:

$$\widetilde{H}^{(l)} = H^{(l)} + \mathbf{1}_T \left(v_c^{(l)}\right)^\top, \qquad l \in \mathcal{K}, \tag{1}$$

where $\mathbf{1}_T \in \mathbb{R}^T$ broadcasts the same vector across all token positions at layer l. The intervention is defined at the activation level and does not update any parameter of f_θ. During training, all candidate-layer vectors are active simultaneously and optimized jointly under a fixed layer-indexed parameterization. After training, source-side validation selects the active layer set and intervention strengths for deployment; vectors outside the selected set are simply not injected at inference time.

For each concept type c, the source supervision set consists of triples $(V_i, Q_i^{(c)}, Y_i^{(c)})$, where $Q_i^{(c)}$ is a concept-targeted prompt and $Y_i^{(c)}$ is the target continuation constructed for that example. The concept vectors are learned by minimizing the generative objective

$$\left\{v_c^{(l)*}\right\}_{l\in\mathcal{K}} = \underset{\{v_c^{(l)}\}_{l\in\mathcal{K}}}{\arg\min}\, \mathbb{E}_{(V,Q^{(c)},Y^{(c)})\sim\mathcal{D}_c} \left[\mathcal{L}_{\text{gen}}\left(Y^{(c)} \mid V, Q^{(c)}; \theta, \{v_c^{(l)}\}_{l\in\mathcal{K}}\right)\right]. \tag{2}$$

Because the same set of concept vectors is optimized across examples with different videos and target expressions, it must support the reconstruction of diverse instances of the corresponding concept type. This shared optimization encourages the vectors to capture semantic information that recurs across the training examples, rather than information specific to an individual video or target expression. Unlike contrastive direction extraction, the vectors are optimized directly through the generative objective and do not require positive–negative activation pairs.

The construction of concept-targeted prompts, target continuations, and the answer-only loss is described in the following subsection. The learned concept vectors remain external to the frozen backbone and can be stored independently, reused across tasks, scaled at inference time, or composed with other concept vectors.

3.2 Concept-Targeted Generative Supervision

As illustrated in Figure 1(d), ConceptTune learns a shared concept vector through concept-targeted generative completion, optimizing the vector while keeping the MLLM backbone frozen. For concept type c, let

$$X^{(c)} = (x_1, \ldots, x_N) = \left[Q^{(c)}; Y^{(c)}\right]$$

denote the tokenized prompt–continuation sequence, where $Q^{(c)}$ is the concept-targeted prompt and $Y^{(c)}$ is the designated target continuation. Given the intervened hidden states, ConceptTune minimizes the answer-only autoregressive loss

$$\mathcal{L}_{\text{gen}}^{(c)} = -\sum_{t=1}^{N} m_t^{(c)} \log p_{\theta,\mathbf{v}_c}\left(x_t \mid V, x_{<t}\right), \tag{3}$$

where $m_t^{(c)} \in \{0,1\}$ equals one only when x_t belongs to the designated target continuation, and $\mathbf{v}_c = \{v_c^{(l)}\}_{l\in\mathcal{K}}$. Prompt tokens are assigned zero weight, whereas target continuation tokens receive unit weight. The loss therefore focuses the learning signal on the content that instantiates the corresponding concept type. Since θ remains frozen, gradients are applied only to the external concept vectors $\mathbf{v}_c$.

The prompt and target continuation are constructed according to the concept type being learned. For action concepts, an action phrase is masked in the prompt and the original phrase serves as the target continuation. For object concepts, an object phrase is masked and reconstructed in the same manner. For temporal concepts, the prompt specifies an event and the target continuation indicates its coarse temporal position in the video: beginning, middle, or end. Across examples, the specific target expressions vary, while the same concept vector is shared and optimized for the corresponding concept type.

All concept vectors are thus learned through the native generative interface of the MLLM rather than through the structured outputs of a particular downstream task. The resulting supervision directly optimizes the vectors to support concept-relevant completion without requiring manually constructed positive–negative activation pairs. After training, the vectors can be applied to downstream tasks whose output formats differ from the source completion objective.

3.3 Hyperparameter Configuration and Test-Time Reuse

After learning the concept vectors, ConceptTune determines their deployment configuration using a disjoint validation split of the source dataset. Let $\mathcal{K}_c^\star \subseteq \mathcal{K}$ denote the selected intervention layers for concept type c, and let $\alpha_{c,l}$ denote the intervention strength at layer l. During source-side configuration selection, the learned vectors remain fixed; only the intervention layers and strengths are selected according to performance on the source validation split. This selection is a deployment-stage choice applied after vector learning; the selected vectors are not re-optimized in isolation after layer selection.

For a target input, the selected concept vector is injected into the residual stream as

$$\widetilde{H}^{(l)} = H^{(l)} + \alpha_{c,l}\mathbf{1}_T\left(v_c^{(l)}\right)^\top, \qquad l \in \mathcal{K}_c^\star. \tag{4}$$

All remaining computations are performed by the original frozen MLLM. Once $\mathcal{K}_c^\star$ and $\{\alpha_{c,l}\}_{l\in\mathcal{K}_c^\star}$ are selected, they remain fixed across target datasets and task formats. No target annotations are used to learn the concept vectors or to select the intervention layers and strengths.

This protocol separates concept-vector learning and configuration selection from target-task evaluation. The learned vectors can therefore be applied to temporal moment retrieval, highlight detection, and video question answering through each task's native prompt and output interface, without updating the backbone, retraining the vectors, or tuning their intervention configurations on the target task. We additionally investigate multi-concept composition in Appendix E.

4 Experiments

4.1 Experimental Setup

Models and concepts. We evaluate ConceptTune on Qwen3-VL-8B (Bai et al., 2025), Qwen3.5-9B (Qwen Team, 2026), and LLaVA-OV1.5-8B (An et al., 2025), learning external vectors for action, object, and coarse temporal-position concepts while keeping each backbone frozen. Implementation details are provided in Appendix A.

Datasets and metrics. We evaluate temporal moment retrieval on Charades-STA (Gao et al., 2017) and ActivityNet-Caption (Krishna et al., 2017) using recall at IoU thresholds $\{0.3, 0.5, 0.7\}$ and mIoU, and highlight detection on QVHighlights (Lei et al., 2021) using mAP and Hit@1. Cross-task transfer is evaluated on VideoMME (Fu et al., 2025) and MVBench (Li et al., 2024) using Qwen3-VL-8B, reporting category-level and overall accuracy.

Source-to-target protocol. Concept vectors are learned from Charades-STA source supervision. Intervention layers and strengths are selected on a disjoint source-validation split and fixed across target datasets and task formats. Target examples and annotations are not used for vector learning or configuration selection. Appendices B and C provide the supervision templates and selection procedure.

Baselines and decoding. We compare with the frozen backbone, contrastive RepE, LoRA, and LOFIT. RepE is evaluated for action and object concepts on Qwen3-VL-8B; LoRA and LOFIT use the same source completion objective and answer-only loss as ConceptTune. Baseline implementations are detailed in Appendix D. Unless otherwise specified, all methods share video sampling and decoding settings, with temperature 0.01, top-$k = 1$, and top-$p = 0.001$.

4.2 Main Results on Video Grounding

Table 1 reports action- and object-targeted transfer across three video MLLMs. With source-selected intervention configurations fixed, ConceptTune improves all reported temporal grounding and highlight detection metrics over the corresponding frozen backbones.

On Qwen3-VL-8B, the action vector improves Charades-STA Recall@0.5 and Recall@0.7 by 5.8 and 5.3 points, respectively, with additional gains on ActivityNet-Caption and QVHighlights. Improvements also hold for the stronger Qwen3.5-9B backbone and LLaVA-OV1.5-8B, where both concept types substantially improve localization on Charades-STA. These results demonstrate consistent cross-benchmark reuse of the learned concept vectors across the three evaluated backbones.

4.3 Comparison with RepE, LoRA, and LOFIT

We compare ConceptTune with contrastive RepE (Zou et al., 2023) and with the parameter-efficient methods LoRA (Hu et al., 2022) and LOFIT (Yin et al., 2024). The RepE comparison examines the direction learning signal under the same target evaluation setting, whereas the LoRA and LOFIT comparison uses matched source supervision to compare different adaptation interfaces.

Comparison with contrastive RepE. Table 2 compares ConceptTune with RepE (Zou et al., 2023) on Qwen3-VL-8B. ConceptTune outperforms RepE on all four metrics for both concept types. Action vectors improve Charades-STA and ActivityNet-Caption mIoU over RepE by 2.6 and 3.9 points, respectively; object vectors improve them by 0.9 and 2.2 points. These results support concept-targeted generative learning as an effective alternative to contrastive direction extraction.

Comparison under matched supervision. Table 3 compares ConceptTune with LoRA (Hu et al., 2022) and LOFIT(Yin et al., 2024) under the same source completion objective. Implementation details are provided in Appendix D. ConceptTune achieves the highest mIoU in all four backbone–concept settings, outperforming the best-performing baseline by 3.2 and 1.9 points for action and object concepts on Qwen3-VL-8B, and by 1.3 and 1.1 points on LLaVA-OV1.5-8B. A full-description supervision control is provided in Appendix F.3, where generic event-description supervision does not reproduce the gains obtained with concept-targeted completion.

Response and strength analysis. We evaluate action and object completion probes on the same 50 videos using Qwen3-VL-8B. For method m, injected vector type c, and probe type p, we compute the change in target-token-averaged log probability relative to the frozen backbone and its within-probe normalization:

$$\Delta_{m,c,p} = \frac{1}{N}\sum_{i=1}^{N}\left(\ell_{i,m,c,p} - \ell_{i,\text{frozen},p}\right), \qquad \widehat{\Delta}_{m,c,p} = \frac{\Delta_{m,c,p}}{\max_{m',c'}|\Delta_{m',c',p}|}, \tag{5}$$

Table 1: Main results on action- and object-targeted transfer. ConceptTune vectors are learned from source supervision and directly reused for temporal grounding and highlight detection. Values in parentheses denote absolute improvements over the corresponding frozen MLLM. Bold scores indicate improvements over Frozen.

Method	Charades-STA				ActivityNet-Captions				QVHighlights	
	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU	mAP	Hit@1
VTG-LLM (2025)	52.0	33.8	15.7	–	–	–	–	–	16.5	33.5
TimeChat (2024)	47.7	22.9	12.5	30.6	30.2	16.9	8.2	21.8	14.5	23.9
VTimeLLM (2024)	51.0	27.5	11.4	31.2	44.0	27.8	14.3	30.4	–	–
Momentor (2024)	42.9	23.0	12.4	29.3	42.6	26.6	11.6	28.5	7.6	17.0
HawkEye (2024)	50.6	31.4	14.5	33.7	49.1	29.3	10.7	32.7	–	–
NumPro (2025)	63.8	42.0	20.6	41.4	55.6	37.5	20.6	38.8	25.0	37.2
TFVTG (2024)	67.0	50.0	24.3	44.5	49.3	27.0	13.4	34.1	–	–
T2SGrid (2026)	71.4	47.0	20.7	44.9	44.8	26.8	16.1	32.5	–	–
Trace (2024)	–	40.3	19.4	38.7	–	37.7	24.0	39.0	26.8	42.7
Time-R1 (2025)	78.1	60.8	35.3	–	58.6	39.0	21.4	–	–	–
Qwen3-VL-8B										
Frozen	70.7	48.6	22.2	45.9	48.8	33.0	20.0	35.6	35.0	54.3
+ConceptTune (Action)	**74.1** (+3.4)	**54.4** (+5.8)	**27.5** (+5.3)	**49.2** (+3.3)	**53.8** (+5.0)	**36.2** (+3.2)	**22.4** (+2.4)	**39.0** (+3.4)	**37.4** (+2.4)	**61.7** (+7.4)
+ConceptTune (Object)	**72.7** (+2.0)	**53.4** (+4.8)	**25.1** (+2.9)	**47.8** (+1.9)	**49.5** (+0.7)	**33.3** (+0.3)	**20.2** (+0.2)	**36.1** (+0.5)	**37.3** (+2.3)	**60.1** (+5.8)
Qwen3.5-9B										
Frozen	78.8	63.8	39.4	55.5	55.9	38.8	23.8	40.6	36.4	60.2
+ConceptTune (Action)	**80.0** (+1.2)	**65.9** (+2.1)	**41.4** (+2.0)	**56.6** (+1.1)	**56.8** (+0.9)	**39.6** (+0.8)	**24.6** (+0.8)	**41.2** (+0.6)	**38.2** (+1.8)	**61.8** (+1.6)
+ConceptTune (Object)	**79.7** (+0.9)	**65.3** (+1.5)	**40.1** (+0.7)	**56.3** (+0.8)	**56.9** (+1.0)	**39.5** (+0.7)	**24.5** (+0.7)	**41.2** (+0.6)	**37.7** (+1.3)	**62.7** (+2.5)
LLaVA-OV1.5-8B										
Frozen	38.6	15.3	5.4	23.5	48.0	28.0	16.0	33.8	15.3	18.1
+ConceptTune (Action)	**41.8** (+3.2)	**26.1** (+10.8)	**11.6** (+6.2)	**26.9** (+3.4)	**49.2** (+1.2)	**31.2** (+3.2)	**18.5** (+2.5)	**35.1** (+1.3)	**16.3** (+1.0)	**19.2** (+1.1)
+ConceptTune (Object)	**41.7** (+3.1)	**22.9** (+7.6)	**10.0** (+4.6)	**26.8** (+3.3)	**49.3** (+1.3)	**29.3** (+1.3)	**17.0** (+1.0)	**36.3** (+2.5)	**16.4** (+1.1)	**19.3** (+1.2)

Table 2: Comparison between contrastive RepE and ConceptTune on Qwen3-VL-8B. Directions are evaluated on the same temporal grounding and highlight detection benchmarks. Values in parentheses denote changes relative to the frozen backbone. Best results within each concept are bold.

Concept	Method	Charades-STA mIoU	ActivityNet-Caption mIoU	QVHighlights mAP	QVHighlights Hit@1
Action	Frozen	45.9	35.6	35.0	54.3
	RepE	46.6 (+0.7)	35.1 (-0.5)	36.0 (+1.0)	58.3 (+4.0)
	ConceptTune	**49.2** (+3.3)	**39.0** (+3.4)	**37.4** (+2.4)	**61.7** (+7.4)
Object	Frozen	45.9	35.6	35.0	54.3
	RepE	46.9 (+1.0)	33.9 (-1.7)	36.4 (+1.4)	57.8 (+3.5)
	ConceptTune	**47.8** (+1.9)	**36.1** (+0.5)	**37.3** (+2.3)	**60.1** (+5.8)

where $N = 50$ and ℓ is computed with teacher forcing. The normalization denominator is shared across methods and vectors within each probe type. Figure 3 (left) reports responses at layers 0 and 35 with $\alpha = 5$. ConceptTune produces the largest matched-concept responses among the evaluated methods, while its positive off-diagonal responses indicate benefits to related completions.

Figure 3 (right) varies ConceptTune's intervention strength over $\alpha \in \{-5, -3, -1, 0, 1, 3, 5\}$ for the matched vector–probe pairs, keeping vectors, layers, and videos fixed. Negative strengths re-

Table 3: Matched-supervision comparison on Charades-STA. LoRA, LOFIT, and ConceptTune use the same source masked-completion objective, answer-only loss, video sampling procedure, and evaluation protocol. Values are mIoU percentages. Best results in each row are bold.

Backbone	Concept	Frozen	LoRA	LOFIT	ConceptTune
Qwen3-VL-8B	Action	45.9	45.1	46.0	**49.2**
Qwen3-VL-8B	Object	45.9	45.9	45.5	**47.8**
LLaVA-OV1.5-8B	Action	23.5	22.6	25.6	**26.9**
LLaVA-OV1.5-8B	Object	23.5	25.4	25.7	**26.8**

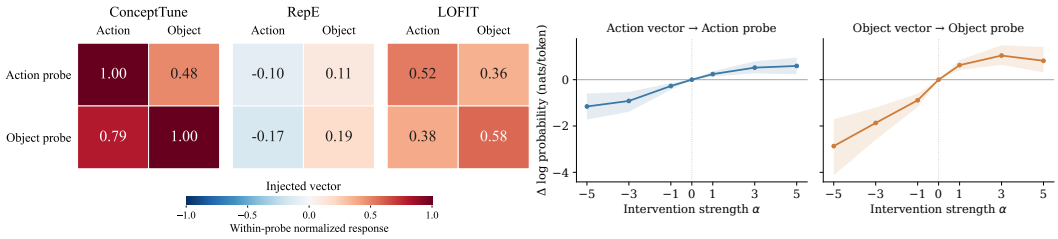


Figure 3: Response and strength analysis of activation interventions on Qwen3-VL-8B. Left: within-probe normalized responses on action and object completion probes for ConceptTune, RepE, and LOFIT. Right: unnormalized responses of the matched vector–probe pairs as the intervention strength α varies. Shaded regions indicate pointwise 95% bootstrap intervals over 50 videos.

duce target likelihood, whereas positive strengths increase it. The action response levels off at larger positive strengths, while the object response peaks at $\alpha = 3$. Curves show unnormalized responses with pointwise 95% video-bootstrap intervals. This diagnostic does not alter the benchmark configurations; further details appear in Appendix F.

A video-permutation control further examines whether the intervention gain depends on matching visual evidence; details are provided in Appendix F.4.

4.4 Cross-Task Transfer to Video Question Answering

We next evaluate whether concept vectors learned from grounding supervision can be reused for video question answering without VQA-specific training. We directly apply the action and object concept vectors learned on grounding data to Qwen3-VL-8B and evaluate them on VideoMME and MVBench. Unlike temporal grounding, these benchmarks require answers in a question-answering format rather than temporal spans. Table 4 reports the category-level and overall accuracies.

For the action concept, ConceptTune improves all six listed category-level scores over the frozen backbone. On VideoMME, the largest gain is observed for Temporal Perception, which increases from 63.64 to 70.91, while the overall accuracy increases from 61.52 to 62.26. On MVBench, Fine-Grained Action, Unexpected Action, and Action Localization improve by 2.00, 2.50, and 2.00 points, respectively.

The object concept also improves category scores. On MVBench, Object Existence increases from 77.50 to 81.00, Object Shuffle from 35.00 to 41.50, and Episodic Reasoning from 48.50 to 50.50. On VideoMME, Object Reasoning, Object Recognition, and Counting also improve over the frozen backbone. ConceptTune outperforms RepE in every listed category for both concepts.

These results demonstrate cross-task transfer from grounding supervision to action- and object-related VideoQA categories without VQA-specific training.

4.5 Efficiency and Parameterization

ConceptTune stores external concept vectors and leaves the MLLM backbone unchanged. Table 5 summarizes the persistent state and direction-construction time measured on the Qwen3-VL-8B

Table 4: Cross-task transfer to video question answering on Qwen3-VL-8B. Action and object concept vectors are learned from grounding supervision and directly reused without VQA-specific training. Values are accuracy percentages; "All" denotes overall benchmark accuracy. Best results within each concept are bold.

Action concept								
Method	VideoMME				MVBench			
	Action Reasoning	Action Recognition	Temporal Perception	All	Fine-Grained Action	Unexpected Action	Action Localization	All
Frozen	55.09	58.15	63.64	61.52	44.00	72.00	39.00	58.10
RepE	52.63	57.19	65.45	60.70	40.50	70.00	37.50	58.40
ConceptTune	**55.44**	**59.11**	**70.91**	**62.26**	**46.00**	**74.50**	**41.00**	**58.70**
Object concept								
Method	VideoMME				MVBench			
	Object Reasoning	Object Recognition	Counting	All	Object Existence	Object Shuffle	Episodic Reasoning	All
Frozen	59.91	64.69	39.18	**61.52**	77.50	35.00	48.50	58.10
RepE	58.81	64.97	39.55	60.67	79.00	37.50	47.50	58.70
ConceptTune	**60.79**	**65.25**	**40.30**	61.48	**81.00**	**41.50**	**50.50**	**58.83**

Table 5: Efficiency comparison for direction construction on Qwen3-VL-8B using 10,000 source examples. Persistent state refers to the data retained or stored during direction construction; the construction time is measured for one run.

Method	Construction time (s)	Persistent state
RepE	3,284.49	5.9 GB activation bank
ConceptTune	2,617.77	0.29 MB concept vectors

action setting with 10,000 source examples. The storage comparison concerns the state retained for direction construction, while the time comparison is based on a single run.

ConceptTune stores one vector per candidate layer, giving a persistent state of $O(Ld)$. With $L = 36$, hidden width $d = 4096$, and bf16 storage, this corresponds to approximately 0.29 MB. By contrast, retaining positive and negative activations for RepE requires an activation bank of $O(2NLd)$, estimated at approximately 5.9 GB under the same dimensions and $N = 10{,}000$ examples. In the same measurement, ConceptTune takes 2,617.77 seconds to construct the directions, compared with 3,284.49 seconds for RepE. In this setting, ConceptTune reduces direction-construction time by approximately 20.3% relative to RepE.

5 Conclusion

We presented **ConceptTune**, a hybrid approach that combines the optimization strength of fine-tuning with the lightweight deployment interface of activation intervention. ConceptTune keeps the video MLLM backbone frozen and optimizes a shared, reusable activation offset directly in the residual stream through concept-targeted generative completion. This shared supervision encourages the vector to capture semantic information common across examples while keeping the learned representation external to the model parameters.

Once the source-side intervention configuration is selected, the learned concept vectors can be reused across datasets and task formats without target-specific training. Experiments on three video MLLMs show improvements in temporal grounding and highlight detection, with grounding-trained vectors also transferring to VideoQA. Comparisons with LoRA, LOFIT, and RepE support the value of concept-targeted learning with activation-level intervention, while independently learned vectors can be composed for multi-concept enhancement.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript writing and editing, retrieval and organization of related literature, experimental code development and debugging, experimental design and analysis, organization, interpretation, and summarization of experimental results, and figure drafting. As described in Appendix B, a language model was also used to extract action and object phrases from existing dataset annotations to construct concept-targeted supervision. All reported experimental results were obtained from model runs. The authors reviewed and tested AI-assisted code and checked AI-assisted text, citations, figures, experimental interpretations, and result summaries against the relevant sources and experimental records. The authors take full responsibility for the final content of this work, including all claims, analyses, code, and other artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work uses publicly available video understanding benchmarks and does not involve collecting new data, human-subject experiments, or interaction with individual participants. The experiments study representation-level interventions for improving video understanding and do not introduce a deployment system for making decisions about people. We follow the intended research use of the benchmark datasets and rely on the licenses and usage conditions provided by their original creators. Potential limitations related to dataset bias, representational imbalance, and model-generated errors remain applicable to the underlying benchmarks and video MLLMs.

REPRODUCIBILITY STATEMENT

The ConceptTune objective, activation intervention, and source-side configuration protocol are described in Sections 3.1–3.3, with multi-concept composition detailed in Appendix E. Section 4.1 specifies the evaluated models, datasets, metrics, baselines, decoding settings, and source-to-target evaluation protocol. The main experimental results are reported in Tables 1–5 and Figure 3, while supervision templates, implementation details, and additional analyses are provided in the appendix. The training and evaluation code, configuration files, and learned concept-vector checkpoints will be made publicly available upon acceptance, subject to the licenses and usage conditions of the underlying models and datasets.

REFERENCES

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Chunsheng Wu, Huajie Tan, Chunyuan Li, Jing Yang, Jie Yu, Xiyao Wang, Bin Qin, Yumeng Wang, Zizhen Yan, Ziyong Feng, Ziwei Liu, Bo Li, and Jiankang Deng. Llava-onevision-1.5: Fully open framework for democratized multimodal training. In *arXiv*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Jinhe Bi, Yujun Wang, Haokun Chen, Xun Xiao, Artur Hecker, Volker Tresp, and Yunpu Ma. Llava steering: Visual instruction tuning with 500x fewer parameters through modality linear representation-steering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15230–15250, 2025.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *CVPR*, 2025.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pp. 5267–5275, 2017.

Chaohong Guo, Yihan He, Yongwei Nie, Fei Ma, Xuemiao Xu, and Chengjiang Long. T2sgrid: Temporal-to-spatial gridification for video temporal grounding. *arXiv preprint arXiv:2603.06973*, 2026.

Yongxin Guo, Jingyu Liu, Mingda Li, Qingbin Liu, Xi Chen, and Xiaoying Tang. Trace: Temporal grounding video llm via causal event modeling. *arXiv preprint arXiv:2410.05643*, 2024.

Yongxin Guo, Jingyu Liu, Mingda Li, Dingxin Cheng, Xiaoying Tang, Dianbo Sui, Qingbin Liu, Xi Chen, and Kevin Zhao. Vtg-llm: Integrating timestamp knowledge into video llms for enhanced video temporal grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3302–3310, 2025.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. Vtimellm: Empower llm to grasp video moments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14271–14280, 2024.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pp. 706–715, 2017.

Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. *Advances in Neural Information Processing Systems*, 34:11846–11858, 2021.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 3045–3059, 2021.

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*, pp. 19730–19742. PMLR, 2023a.

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, 2023b.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22195–22206, 2024.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, 2021.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. Tempcompass: Do video llms really understand videos? In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 8731–8772, 2024.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681*, 2023.

Long Qian, Juncheng Li, Yu Wu, Yaobo Ye, Hao Fei, Tat-Seng Chua, Yueting Zhuang, and Siliang Tang. Momentor: Advancing video large language model with fine-grained temporal reasoning. *arXiv preprint arXiv:2402.11435*, 2024.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. Timechat: A time-sensitive multimodal large language model for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14313–14323, 2024.

Anushka Sivakumar, Andrew Zhang, Zaber Hakim, and Chris Thomas. Steervlm: Robust model control through lightweight activation steering for vision language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23640–23665, 2025.

Jingran Su, Jingfan Chen, Hongxin Li, Yuntao Chen, Li Qing, and Zhaoxiang Zhang. Activation steering decoding: Mitigating hallucination in large vision-language models through bidirectional hidden state intervention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12964–12974, 2025.

Ye Wang, Ziheng Wang, Boshen Xu, Yang Du, Kejun Lin, Zihan Xiao, Zihao Yue, Jianzhong Ju, Liang Zhang, Dingyi Yang, et al. Time-r1: Post-training large vision language model for temporal video grounding. *arXiv preprint arXiv:2503.13377*, 2025.

Yueqian Wang, Xiaojun Meng, Jianxin Liang, Yuxuan Wang, Qun Liu, and Dongyan Zhao. Hawkeye: Training video-text llms for grounding text in videos. *arXiv preprint arXiv:2403.10228*, 2024.

Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Zhu JianHao, Cenyuan Zhang, Xiaoqing Zheng, and Xuan-Jing Huang. Advancing parameter efficiency in fine-tuning via representation editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13445–13464, 2024a.

Yongliang Wu, Xinting Hu, Yuyang Sun, Yizhou Zhou, Wenbo Zhu, Fengyun Rao, Bernt Schiele, and Xu Yang. Number it: Temporal grounding videos like flipping manga. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 13754–13765, 2025.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962, 2024b.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9777–9786, 2021.

Fangcong Yin, Xi Ye, and Greg Durrett. Lofit: Localized fine-tuning on llm representations. *Advances in neural information processing systems*, 37:9474–9506, 2024.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, 2022.

Minghang Zheng, Xinhao Cai, Qingchao Chen, Yuxin Peng, and Yang Liu. Training-free video temporal grounding using large-scale pre-trained models. In *European Conference on Computer Vision*, pp. 20–37. Springer, 2024.

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.

A Implementation Details

Models and hardware. We evaluate ConceptTune on three video-capable multimodal language models: Qwen3-VL-8B, Qwen3.5-9B, and LLaVA-OV1.5-8B. The backbone parameters, including the visual encoder and language model, remain frozen throughout concept-vector learning and downstream inference. All experiments are conducted on two NVIDIA RTX 3090 GPUs, each with 24GB of memory. We use PyTorch 2.5.1 with the CUDA 11.8 build and Transformers 4.57.1. Model computations and concept vectors use bfloat16 unless otherwise specified.

Concept-vector parameterization. For each concept c and candidate Transformer layer l, ConceptTune maintains one external vector $v_c^{(l)} \in \mathbb{R}^d$, where d is the hidden width of the corresponding backbone. The vector is shared across all training examples associated with the same concept. It is added to the residual-stream output of the selected layer and broadcast across all token positions. No parameter of the backbone is updated.

Optimization. Concept vectors are learned from the source supervision set using the concept-targeted generative objective described in Section 3. Action and object concepts use masked multimodal completion, whereas temporal concepts use coarse temporal-position completion. We use AdamW with a learning rate of 5×10^{-4} and train for one epoch. The loss is evaluated only on the designated target continuation; prompt tokens and other non-target tokens receive zero loss weight. Consequently, gradients are propagated only to the external concept vectors.

Video processing. Videos are sampled at 1 frame per second for both concept-vector learning and downstream evaluation, unless otherwise specified by the corresponding benchmark protocol. The same sampling procedure is used for ConceptTune and the controlled baseline comparisons.

Candidate layers and configuration selection. Concept vectors are initialized and optimized for all candidate Transformer layers of each backbone. After concept learning, the active intervention layers and their strengths are selected using a held-out validation subset of the source dataset. The learned vectors remain fixed during this procedure. We search the intervention strength over $\alpha \in \{-5, -4, \ldots, 4, 5\}$. The selected layer set and strengths are then fixed before evaluation on target datasets and task formats. No target annotations or target-test examples are used for vector learning, layer selection, or strength selection.

Inference and decoding. At inference time, the selected concept vectors are injected into the residual stream of the frozen backbone at the source-selected layers:

$$\widetilde{H}^{(l)} = H^{(l)} + \alpha_{c,l} \mathbf{1}_T \left(v_c^{(l)}\right)^\top.$$

The intervention is applied without updating the backbone or the concept vectors. Unless otherwise specified, generation uses temperature 0.01, top-$k = 1$, and top-$p = 0.001$.

Multiple-concept inference. When multiple concepts are applied simultaneously, their orthogonalized vectors are combined using independently selected strengths:

$$\widetilde{H}^{(l)} = H^{(l)} + \mathbf{1}_T \left(\sum_{c \in \mathcal{C}'} \alpha_{c,l} \bar{v}_c^{(l)} \right)^\top,$$

where $\mathcal{C}'$ is the selected concept set and $\bar{v}_c^{(l)}$ denotes the orthogonalized concept vector obtained using the norm-preserving procedure in Appendix E. This post-processing introduces no additional trainable parameters or supervision.

B Concept Supervision and Prompt Templates

This section describes how source annotations are converted into concept-targeted completion examples and provides the prompt templates used for concept learning and downstream inference.

B.1 Constructing Concept Supervision

We construct source supervision from video grounding annotations in Charades-STA (Gao et al., 2017). Each annotation provides a video, an event description, and the corresponding temporal interval. For each concept type, we construct a training example $(V, Q^{(c)}, Y^{(c)})$, where V is the video, $Q^{(c)}$ is the concept-targeted prompt, and $Y^{(c)}$ is the target continuation.

For action and object supervision, a language model parses the event description to identify the phrase to be masked. For actions, it is instructed to extract the main action verb or verb phrase. For objects, it extracts the main object noun that the person interacts with, excluding person references such as "person," "man," and "woman." The parser returns the masked description and the extracted phrase in JSON format, and the results are cached before concept-vector learning. The extracted phrase serves as the completion target.

For temporal supervision, the target is derived directly from the annotated event interval and video duration. Let s and e denote the event start and end times, and let D denote the video duration. We compute the relative event midpoint

$$r = \frac{s+e}{2D}, \tag{6}$$

and assign the target label as

$$Y^{(\text{temporal})} = \begin{cases} \texttt{beginning}, & r < 0.33, \\ \texttt{middle}, & 0.33 \le r < 0.66, \\ \texttt{end}, & r \ge 0.66. \end{cases} \tag{7}$$

The event timestamps are used to construct this label but are not included in the temporal-completion prompt.

B.2 Concept-Learning Prompts

The templates below show the textual prompts used for Qwen3-VL-8B. Each prompt is paired with the video input and formatted using the backbone's chat template. Braced fields denote sample-specific values.

Action completion. The action phrase is replaced with `[MASK]`, while the remaining description provides context. The annotated interval identifies the event to inspect.

> **Prompt:** Watch the video from frame {start} to frame {end}. Fill in the [MASK] in this description based on the video: '{masked description}'. The action verb for the [MASK] is
>
> **Target:** {action phrase}

For example, the description "person close the laptop." becomes "person [MASK] the laptop.", with `close` as the target.

Object completion. The object phrase is masked while the action and remaining context are retained.

> **Prompt:** Watch the video from {start} to {end} seconds. Fill in the [MASK] with the correct object noun in this description based on the video: '{masked description}'. The object noun for the [MASK] is
>
> **Target:** {object phrase}

For the same example, the masked description is "person close the [MASK].", and the target is `laptop`.

The interval fields are populated from the source start and end timestamps. The Qwen3-VL action template retains the wording "frame" shown above. Model-specific variants use "seconds" in the interval instruction or the shorter completion suffix "The [MASK] is", while retaining the same completion target.

Temporal-position completion. The complete event description is retained, and the model predicts its coarse position within the video.

> **Prompt:** Watch the video. The event '{event description}' happens in the [MASK] of the video. The [MASK] is
> **Target:** beginning / middle / end

Each example has one target label determined by Equation 7; the slash-separated labels above list the possible answers rather than a literal target sequence.

B.3 Training Sequence and Loss Masking

The target continuation is appended after the formatted video-text prompt, and the model is trained autoregressively with teacher forcing. Labels corresponding to the prompt prefix, including visual input positions and instructions, are set to the loss-ignore index `-100`. The target continuation and its sequence-termination token are retained for loss computation.

Thus, the objective supervises recovery of the designated phrase or temporal label rather than reconstruction of the prompt. Within each concept type, the same layer-wise concept vectors are optimized across examples with different videos and target continuations. For example, action supervision updates a shared set of action vectors rather than learning a separate vector for each action word.

B.4 Downstream Inference Prompts

At deployment, concept-learning prompts are replaced with the downstream task prompts. The learned vectors and their source-selected intervention configuration remain fixed. Source completion targets are not appended to downstream inputs.

Temporal moment retrieval. For Charades-STA and ActivityNet-Caption, the prompt requests the temporal interval corresponding to the query:

> Accurately pinpoint the event '{query}' in the video, answer in the format: 'From x to y'.

Highlight detection. For QVHighlights, the prompt requests highlight frames and their saliency scores:

> Please find the highlight contents in the video described by the query {query}. Determine the highlight frames and its saliency score on a scale from 1 to 5. If the video content more related to the query, the saliency score should be higher. The output format should be like: 'The highlight frames are in the 80, 81, 82, 83, 84, 85, 86, 87, 88, 89 frames. Their saliency scores are 1.3, 1.5, 2.6, 3.0, 2.9, 4.0, 3.7, 3.2, 2.1, 2.3'.

Video question answering. For the multiple-choice evaluations on MVBench and VideoMME, the question and answer options are inserted into the following template:

> Answer the question: "{question}" according to the content of the video. Select the answer from :{options}. Output your think process within the `<think> </think>` tags. Then, provide your answer within the `<answer> </answer>` tags, output the corresponding letter of the option.

The predicted option letter is extracted from the generated answer for evaluation.

C Source-Side Configuration Selection

ConceptTune separates concept-vector learning from the selection of deployment configurations. During learning, vectors at all candidate layers are optimized jointly. During configuration selection, the learned vectors remain fixed, and only the intervention layers and strengths are varied. Selection is performed separately for each concept type and backbone.

Table 6: Source-selected configurations for single-concept interventions. Each configuration applies to action, object, and temporal concepts. Layer indices are zero-based.

Backbone	Intervention layer	Strength α
Qwen3-VL-8B	0	5
Qwen3.5-9B	7	1
LLaVA-OV1.5-8B	0	4

Configuration space. Let $\mathcal{K}$ denote the candidate layers for which concept vectors are learned. In our experiments, $\mathcal{K}$ contains all language-model Transformer layers: $\{0, \ldots, 35\}$ for Qwen3-VL-8B, $\{0, \ldots, 31\}$ for Qwen3.5-9B, and $\{0, \ldots, 35\}$ for LLaVA-OneVision-1.5-8B. The vectors at all layers in $\mathcal{K}$ are learned jointly, while source-side configuration selection determines the active subset used at inference time. A deployment configuration for concept c is

$$\gamma_c = (\mathcal{K}_c, \{\alpha_{c,l}\}_{l \in \mathcal{K}_c}),$$

where $\mathcal{K}_c \subseteq \mathcal{K}$ is the active layer set and $\alpha_{c,l}$ is the strength applied to the vector at layer l. Layers outside $\mathcal{K}_c$ receive no intervention. For single-layer configurations, we evaluate candidate layers with strengths from $\mathcal{A} = \{-5, -4, \ldots, 4, 5\}$. Positive and negative strengths apply the learned direction with opposite signs, while zero strength recovers the unmodified backbone. Multi-layer configurations specify an active layer set together with its corresponding strengths.

Source-validation criterion. We evaluate candidate configurations using temporal moment retrieval on a fixed source-validation subset of Charades-STA. The model receives the original event query and predicts its temporal interval using the downstream grounding prompt, rather than the masked-completion prompt used for concept learning. Configurations are ranked by mean intersection over union (mIoU) between the predicted and annotated intervals.

Let Γ_c denote the set of evaluated configurations and $\mathcal{D}_{\text{val}}$ the source-validation subset. The selected configuration is

$$\gamma_c^\star = \arg\max_{\gamma \in \Gamma_c} \frac{1}{|\mathcal{D}_{\text{val}}|} \sum_{i \in \mathcal{D}_{\text{val}}} \text{IoU}\left(\widehat{I}_i(\gamma), I_i\right), \tag{8}$$

where $\widehat{I}_i(\gamma)$ is the interval predicted under configuration γ and I_i is the annotated interval. The same validation examples, video preprocessing, task prompt, and decoding settings are used across candidate configurations. Neither the backbone nor the concept vectors are updated during this evaluation.

Fixed reuse across datasets and tasks. After selection, the concept vectors, active layers, and intervention strengths are fixed for target evaluation. The configuration is reused within the same backbone for temporal moment retrieval, highlight detection, and video question answering. Only the task prompt and expected output format change; no target-specific vector training or configuration search is performed. Target annotations are used for reporting evaluation metrics, not for selecting intervention configurations.

Selected configurations. Table 6 summarizes the final source-selected configurations. For each backbone, source-side validation independently evaluates the configurations for the three concept types. The search happens to select the same layer and strength for action, object, and temporal concepts; this equality is not imposed during training or configuration selection. The selected configuration is then fixed across downstream datasets and task formats.

D Baseline Implementation Details

This section describes the implementations of the frozen backbone, contrastive RepE, LoRA, and LOFIT. We distinguish between methods that estimate directions from activation contrasts and methods that optimize trainable components using concept-targeted completion.

D.1 FROZEN BACKBONE

The frozen baseline uses the original pretrained MLLM without weight adaptation or activation intervention. It receives the same downstream task prompts and uses the same evaluation pipeline as the corresponding ConceptTune model. Improvements are measured relative to this backbone under the corresponding benchmark setting.

D.2 CONTRASTIVE REPE

We implement a contrastive direction-extraction baseline based on RepE (Zou et al., 2023) for action and object concepts on Qwen3-VL-8B.

Contrast construction. For each source example, we retain the video and masked event description. The positive description fills the masked position with the original action or object phrase, whereas the negative description fills it with a different phrase of the same concept type. Both descriptions are inserted into the same temporal grounding prompt. Thus, the contrast changes the target phrase while keeping the video and surrounding textual context fixed.

Negative phrases are sampled from source labels occurring at least three times. We preferentially sample a phrase with the same number of whitespace-separated words as the positive phrase; if no such candidate exists, we sample another distinct phrase of the same type. Negative sampling uses a fixed random seed of 42.

Activation extraction. The positive and negative inputs are processed separately by the frozen backbone. At each candidate layer, we collect the hidden representation at the final input-token position. Denote the representations for example i at layer l by $h_{i,+}^{(l)}$ and $h_{i,-}^{(l)}$. Their difference is

$$\delta_i^{(l)} = h_{i,+}^{(l)} - h_{i,-}^{(l)}. \tag{9}$$

We apply principal component analysis to the matrix of paired differences and use its first principal component as the layer-wise steering direction. Its sign is oriented to have a nonnegative inner product with the mean positive–negative difference.

Deployment. The extracted direction is added to the residual-stream representation at the configured layers and broadcast across token positions. Direction extraction does not update the backbone and does not use gradient-based optimization. Our RepE comparisons cover action and object concepts; we do not include a temporal RepE baseline.

D.3 LORA

LoRA (Hu et al., 2022) provides a weight-adaptation baseline trained with concept-targeted completion. For a selected weight matrix W, LoRA parameterizes the update as

$$W' = W + \frac{\alpha_{\mathrm{LoRA}}}{r} BA, \tag{10}$$

where A and B are trainable low-rank matrices and the original weight W remains fixed.

Our saved adapter configurations use rank $r = 8$, scaling parameter $\alpha_{\mathrm{LoRA}} = 16$, and dropout 0.05, targeting the `q_proj` and `v_proj` modules. We train separate adapters for action and object completion. The objective supervises the target continuation while masking the prompt prefix, following the supervision format described in Appendix B.

At evaluation time, the learned adapter is retained while the completion prompt is replaced with the downstream task prompt. No ConceptTune vector is added to the LoRA model.

D.4 LOFIT

LOFIT (Yin et al., 2024) adapts frozen models by learning additive offsets on intermediate representations while keeping the pretrained parameters fixed.

Our LOFIT implementation learns layer-wise additive offsets under concept-targeted completion supervision. We initialize the offsets to zero and optimize them with AdamW using an answer-only loss, training separate checkpoints for action and object concepts. At inference time, the offsets are applied at the configured layers with the specified intervention strengths.

Learning-rate configurations. For LoRA and LOFIT, we evaluate learning rates in $\{1\times10^{-5}, 2\times 10^{-5}, 5\times10^{-5}, 2\times10^{-4}\}$ under action- and object-targeted supervision on Qwen3-VL-8B and LLaVA-OV1.5-8B. For each baseline, backbone, and concept type, we select the learning rate that achieves the highest mIoU on the source validation split and report the corresponding test results.

D.5 Benchmark Comparisons and Response Diagnostics

Benchmark evaluation uses each method's saved adaptation artifacts with the corresponding downstream prompts and decoding settings. LoRA and LOFIT use the same type of action or object completion supervision as ConceptTune, whereas RepE obtains its directions from positive–negative activation contrasts.

For the concept-response diagnostic, ConceptTune, RepE, and LOFIT are evaluated on Qwen3-VL-8B using the same intervention layers, $\{0, 35\}$, and strength, $\alpha = 5$. These settings provide a common intervention configuration for the diagnostic rather than an independently optimized setting for each baseline. They should not be interpreted as the deployment configuration for every benchmark experiment.

E Multi-Concept Composition

ConceptTune learns concept vectors independently. This section describes how multiple vectors can be combined at inference time and evaluates their joint effects on temporal grounding.

E.1 Composition Procedure

Let $\mathcal{C}' \subseteq \mathcal{C}$ denote the concept types selected for composition. At layer l, the combined intervention is

$$\widetilde{H}^{(l)} = H^{(l)} + \mathbf{1}_T \left(\sum_{c \in \mathcal{C}'} \alpha_{c,l} \bar{v}_c^{(l)} \right)^\top, \tag{11}$$

where $\alpha_{c,l}$ is the intervention strength and $\bar{v}_c^{(l)}$ is the concept vector used for composition. For a concept that is inactive at layer l, its strength is set to zero.

To reduce overlap between independently learned directions, we orthogonalize the vectors before combining them. At a layer with m active concepts, we arrange their vectors as columns:

$$V^{(l)} = \left[v_{c_1}^{(l)}, \ldots, v_{c_m}^{(l)} \right] \in \mathbb{R}^{d \times m}. \tag{12}$$

We compute the thin singular value decomposition

$$V^{(l)} = U^{(l)} \Sigma^{(l)} \left(W^{(l)} \right)^\top, \tag{13}$$

and construct the orthogonalized vectors as

$$\overline{V}^{(l)} = U^{(l)} \left(W^{(l)} \right)^\top D^{(l)}, \tag{14}$$

where

$$D^{(l)} = \operatorname{diag} \left(\left\| v_{c_1}^{(l)} \right\|_2, \ldots, \left\| v_{c_m}^{(l)} \right\|_2 \right). \tag{15}$$

The columns of $\overline{V}^{(l)}$ define the vectors $\bar{v}_c^{(l)}$ in Equation 11. This procedure produces mutually orthogonal columns while preserving the original vector norms.

Orthogonalization is applied after concept learning. It introduces no additional trainable parameters or joint training objective, and the backbone remains unchanged during multi-concept inference.

Table 7: Single-concept and orthogonalized pairwise interventions on Charades-STA. All concept vectors are learned independently and combined only at inference time. Values are percentages.

Intervention	R@0.3	R@0.5	R@0.7	mIoU
Qwen3-VL-8B				
Frozen	70.7	48.6	22.2	45.9
Action	74.1	54.4	27.5	49.2
Object	72.7	53.4	25.1	47.8
Temporal	73.6	52.5	25.2	48.1
Action + Object	73.2	54.2	26.3	48.2
Action + Temporal	75.5	56.7	28.8	50.3
Object + Temporal	73.9	54.4	26.6	48.7
LLaVA-OV1.5-8B				
Frozen	38.6	15.3	5.4	23.5
Action	41.8	26.1	11.6	26.9
Object	41.7	22.9	10.0	26.8
Temporal	43.3	20.2	7.5	26.3
Action + Object	43.0	23.8	11.1	27.6
Action + Temporal	43.4	22.2	10.3	27.1
Object + Temporal	44.0	21.5	9.5	27.3

E.2 Experimental Results

We evaluate pairwise concept composition on Charades-STA using Qwen3-VL-8B and LLaVA-OV1.5-8B. Table 7 reports the frozen backbone, individual concept interventions, and pairwise combinations. All pairwise interventions use the orthogonalization procedure above. We report recall at IoU thresholds of 0.3, 0.5, and 0.7, together with mIoU.

Qwen3-VL-8B. Action–temporal and object–temporal composition improve mIoU over their respective constituent interventions, with action–temporal composition achieving the highest mIoU of 50.3. Action–object composition improves over object alone but remains below action alone, indicating that the benefit depends on the concept pair.

LLaVA-OV1.5-8B. All three pairwise combinations improve mIoU over their respective constituent interventions, with action–object composition achieving the highest mIoU of 27.6. These results show that independently learned concept vectors can provide complementary benefits when composed.

Discussion. These results demonstrate that independently learned concept vectors can provide complementary gains when combined. The benefit varies with the backbone and concept pair: action–object composition improves mIoU on LLaVA-OV1.5-8B, but not over the stronger action-only intervention on Qwen3-VL-8B. Orthogonalization removes linear overlap between the vectors at the intervention layer; their joint effect on downstream predictions also depends on the subsequent model computation.

F Additional Analyses

This section supplements the response and strength analysis in the main paper with implementation details, unnormalized results, and a logit-lens-style inspection of the learned action directions.

F.1 Concept-Response Diagnostics

Probe construction. We use a fixed set of 50 distinct Charades-STA videos, with at most one annotated event per video. Action and object probes for each video refer to the same event interval and description. The corresponding action or object phrase is replaced with `[MASK]`, and the original phrase serves as the target continuation. Each masked description contains exactly one masked position.

Table 8: Unnormalized concept responses on 50 videos. Values are changes in mean target-token log probability relative to the frozen backbone, in nats/token, at $\alpha = 5$.

Method	Injected vector	Action probe	Object probe
ConceptTune	Action	0.592	0.649
	Object	0.284	0.816
RepE	Action	-0.060	-0.137
	Object	0.064	0.153
LOFIT	Action	0.310	0.310
	Object	0.212	0.470

The diagnostic prompt is:

> Watch the video from {start} to {end} seconds. Fill in the [MASK] with the correct {phrase type} based on the video: '{masked description}'. Answer only with the missing phrase.

Here, {phrase type} is "action verb" or "object noun." All methods are evaluated on the same videos, prompts, and target continuations.

We use this fixed 50-video set only for the response diagnostic, with videos excluded from the first 10,000 source-training records.

Target-token scoring. We compute the mean log probability of the reference continuation under teacher forcing. For an example with target tokens $Y = (y_1, \ldots, y_M)$, the score is

$$\ell(V, Q, Y) = \frac{1}{M} \sum_{t=1}^{M} \log p(y_t \mid V, Q, y_{<t}). \tag{16}$$

Only tokens belonging to the target phrase are scored. Prompt tokens, chat-template tokens, and end-of-sequence tokens are excluded. Scores use natural logarithms and are reported in nats per target token.

The frozen and intervened conditions share the same tokenized input for each probe. We first compute their score difference for each video and then average these differences across videos. Consequently, every video receives equal weight regardless of the number of tokens in its target phrase. This diagnostic measures reference-continuation likelihood, not free-generation accuracy.

Intervention settings. All three methods use Qwen3-VL-8B with interventions at layers 0 and 35, using zero-based indexing. For the response matrices, the intervention strength is fixed at $\alpha = 5$. Saved vectors are used without norm matching across methods. Thus, a shared coefficient provides the same numerical scaling factor but does not imply equal perturbation norms. The LOFIT implementation is described in Appendix D.

Unnormalized responses. Table 8 reports the values underlying the normalized heatmap in the main paper. Normalization is applied only for visualization and does not modify the vectors or the model outputs.

For the action and object probe types, the shared normalization denominators are approximately 0.5923 and 0.8165, respectively. Each denominator is the maximum absolute mean response over all methods and injected vector types for that probe. The normalized values therefore preserve signs and relative magnitudes within a probe type; values from different probe types should not be interpreted on a common absolute scale.

Within ConceptTune, the action vector produces a larger response than the object vector on action probes, while the object vector produces a larger response on object probes. Both vectors also produce positive cross-concept responses.

Table 9: Control experiment comparing full-description supervision with the frozen backbone and action-targeted ConceptTune on Charades-STA.

Method	R@0.3	R@0.5	R@0.7	mIoU
Frozen	70.70	48.60	22.20	45.90
Full-description supervision	70.43	48.98	22.39	45.76
Action-targeted ConceptTune	74.10	54.40	27.50	49.20

F.2 Intervention-Strength Analysis

We vary the ConceptTune intervention strength over $\alpha \in \{-5, -3, -1, 0, 1, 3, 5\}$ while keeping the vectors, intervention layers, and evaluation videos fixed. The main-paper curves show the matched action-vector/action-probe and object-vector/object-probe responses without normalization. At $\alpha = 0$, no perturbation is applied, and the response relative to the frozen backbone is zero.

Uncertainty estimation. For each strength and matched vector–probe pair, we compute a paired score difference for each of the 50 videos. We then resample these video-level differences with replacement 2,000 times, computing the mean for each resample. The 2.5th and 97.5th percentiles define the pointwise 95% bootstrap interval. These intervals describe variation across the sampled videos, not variation across independently trained vector checkpoints.

Observed behavior. Negative strengths reduce target-continuation likelihood for both matched pairs, whereas positive strengths increase it over the evaluated range. The action response increases from 0.240 at $\alpha = 1$ to 0.520 at $\alpha = 3$ and 0.592 at $\alpha = 5$. The object response increases from 0.627 at $\alpha = 1$ to 1.041 at $\alpha = 3$, then decreases to 0.816 at $\alpha = 5$. Thus, larger positive strengths do not necessarily yield larger responses. This sweep characterizes the effect of scaling a fixed learned direction.

F.3 Full-Description Supervision Control

To examine whether the observed transfer gains arise simply from training a vector to generate event descriptions, we train a control vector using the same source examples, vector parameterization, optimization budget, and video preprocessing as ConceptTune. The only change is the supervision target: instead of reconstructing the masked action phrase, the control is trained to generate the complete original event description.

The control vector is evaluated on Charades-STA using the same Qwen3-VL-8B backbone, layer, intervention strength, and downstream evaluation protocol. As shown in Table 9, full-description supervision remains close to the frozen backbone, obtaining an mIoU of 45.76 compared with 45.90 for the frozen model. In contrast, the action-targeted ConceptTune vector reaches an mIoU of 49.20. The control therefore suggests that generic event-description supervision is insufficient for learning a reusable action-oriented intervention, whereas concept-targeted completion provides a more effective learning signal.

F.4 Video-Evidence Dependence Control

The concept-response analysis in Figure 3 measures how the learned vectors affect concept-targeted completion. To examine whether this effect depends on matching visual evidence, we perform a video-permutation control without retraining the vectors. We use the same 50 held-out videos, prompts, target continuations, intervention layers, and intervention strength as in the response analysis. For each example, we evaluate two video conditions: the original matching video and a randomly permuted video from another example. The frozen backbone and the intervened model are evaluated under both conditions.

For each example, we compute the mean target-token log probability under teacher forcing. The intervention gain is defined as the difference between the intervened and frozen scores:

$$\Delta_i^{\text{cond}} = \ell_{i,\text{vector}}^{\text{cond}} - \ell_{i,\text{frozen}}^{\text{cond}}, \qquad \text{cond} \in \{\text{matched}, \text{shuffled}\}. \tag{17}$$

Table 10: Video-evidence dependence control on 50 held-out videos. Values report the mean gain in target-token log probability relative to the frozen backbone, in nats/token. The final column reports the paired difference between the matched and shuffled conditions with a 95% bootstrap interval.

Concept	Matched video	Shuffled video	Matched − Shuffled
Action	0.59	0.26	0.33 [0.23, 0.49]
Object	0.82	0.07	0.75 [0.30, 1.39]

Table 11: Top eight decoded entries from the vocabulary projection of action directions at layer 35 of Qwen3-VL-8B. Entries are shown in ranked order, preserving capitalization.

Method	Decoded entries
ConceptTune	takes, runs, opening, throws, Opening, puts, UND, Runs
RepE	There, THERE, OutOf, anim, ToSelector,),], Intialized

Table 10 summarizes the results. With the original matching videos, the action and object vectors improve the target-token log probability by 0.59 and 0.82 nats/token, respectively. After permuting the videos across examples, the corresponding gains decrease to 0.26 and 0.07. The paired differences between the matched and shuffled conditions are 0.33 for action and 0.75 for object, with 95% bootstrap intervals of $[0.23, 0.49]$ and $[0.30, 1.39]$, respectively. The intervals are obtained by resampling videos rather than individual target tokens.

These results indicate that the intervention gains are amplified when the video matches the target completion. The remaining positive gains under permuted videos also suggest that the learned vectors may affect language-side completion preferences in addition to exploiting visual evidence. Thus, the control supports a dependence on matching video evidence without implying that the learned vectors encode purely visual or perfectly disentangled concepts.

F.5 Vocabulary Projection of Action Directions

Projection procedure. We inspect the lexical associations of the saved action directions using a vocabulary projection inspired by the logit lens. For a direction $v^{(l)} \in \mathbb{R}^d$, we compute

$$z^{(l)} = W_{\text{out}} v^{(l)}, \tag{18}$$

where W_{out} is the output embedding matrix of Qwen3-VL-8B. We rank vocabulary tokens by their projection scores and decode the highest-ranked token IDs.

We directly project each saved direction through the model's output embedding matrix, without additional normalization, and inspect the resulting token rankings.

Decoded tokens. Table 11 lists the eight highest-ranked decoded entries in the saved final-layer diagnostic. ConceptTune's action direction places several recognizable action words near the top of the ranking, including "takes," "runs," "opening," and "throws." The corresponding RepE ranking contains a mixture of general words and token fragments.

The decoded tokens vary across layers. At the final layer, ConceptTune exhibits clear action-related vocabulary alignment, complementing the input-conditioned response analysis.

G Limitations

ConceptTune currently relies on source-side validation to select intervention layers and strengths. Although the selected configuration is reused across downstream datasets and task formats, this selection introduces additional evaluation cost. Developing more automatic configuration-selection criteria is an important direction for future work.

The learned concept vectors are specific to the backbone in which they are optimized. Our experiments demonstrate reuse across datasets and tasks within the same backbone, rather than direct

transfer of a vector between different model families. Transferring concept representations across backbones remains an open problem.

Our experiments focus on action, object, and coarse temporal-position concepts. Extending the framework to richer semantic relations and improving concept selectivity are natural directions for future work.