SUPPRESSING THE SHORTCUT HIDDEN BEHIND THE GAINS OF VIDEOLLM FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised fine-tuning (SFT) is the stage in which Video Large Language Models (VideoLLMs) advance their capability to reason over video. However, the VideoLLM also learns to answer from textual regularities of its training data, a shortcut that bypasses the video. The regularities behind the shortcut are hard to define at the data level, and the shortcut forms regardless of training epoch, data size, or backbone. The shortcut is hidden behind a rise in accuracy within the training distribution, yet it is not required for the rise and even costs SFT the gain the video offers under distribution shift. We therefore propose Video-Evidence Training via Orthogonal projection (VETO), which intervenes on the fine-tuning update where the shortcut is written rather than on the data. VETO calibrates the subspace that the text alone occupies in the activations of the base model and projects the update onto its orthogonal complement. The VideoLLM is thus left to ground its answer in the full input, with the shortcut suppressed as it forms and never defined. Across seven VideoLLMs VETO preserves the gain of SFT on the training distribution and suppresses the shortcut. On six benchmarks under shift it gains more than SFT and answers from the full input rather than the shortcut.

1 INTRODUCTION

Video Large Language Models (VideoLLMs) extend the reasoning that Large Language Models (LLMs) acquire from text to video, an input of rich, dynamic visual semantics (Bai et al., 2025b;a; Li et al., 2025; Clark et al., 2026; Team et al., 2026). Supervised fine-tuning (SFT) remains the established stage for adapting a pretrained VideoLLM to a downstream task (Bai et al., 2025b) and grounding that reasoning in the video. Even recent methods trained with reinforcement learning (RL) retain this stage as the basis of their improvement (Wang et al., 2025a;b). While SFT is thus a core element in advancing VideoLLMs, it is prone to learning superficial correlations rather than reading the modality it is given.

These correlations, which we call shortcuts (Geirhos et al., 2020; Dancette et al., 2021), are regularities of the training set that predict the label within the training distribution but may not hold outside it. Video QA is especially susceptible, as many questions can be answered from regularities in the text alone (Yang et al., 2025), and SFT readily comes to rely on them (Li et al., 2023; Chen & Li, 2026). Existing approaches counter a shortcut by filtering it from the data (Chen et al., 2020; Shvetsova et al., 2025), penalizing it in the loss (Cadene et al., 2019; Clark et al., 2019; Niu et al., 2021; Chen & Li, 2026), or adjusting the trained model or its decoding afterward (Wang et al., 2024; Yang et al., 2025). Filtering has to define which regularity counts as the shortcut before it can act (Dancette et al., 2021), while the regularities fine-tuning will pick up are rarely known in advance, and the loss and the trained model constrain what the model outputs rather than the update through which the shortcut is written. A remedy would instead have to act where the shortcut forms, which raises our central question: *where does the shortcut form in the course of fine-tuning, and can fine-tuning be constrained there to suppress it?*

To address this question, we track how the reliance of a VideoLLM on the text changes over SFT, both within the training distribution and under distribution shift. We fine-tune on filtered video QA data where the text alone gives the base model little ground for its answer. After training, the VideoLLM answers far more questions from the text alone than the base model does, regardless of training epoch, data size, or backbone. Within the training distribution, this may seem to cost nothing, since full-input accuracy rises alongside it, as Fig. 1(a) shows. Under the shift, however,

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.

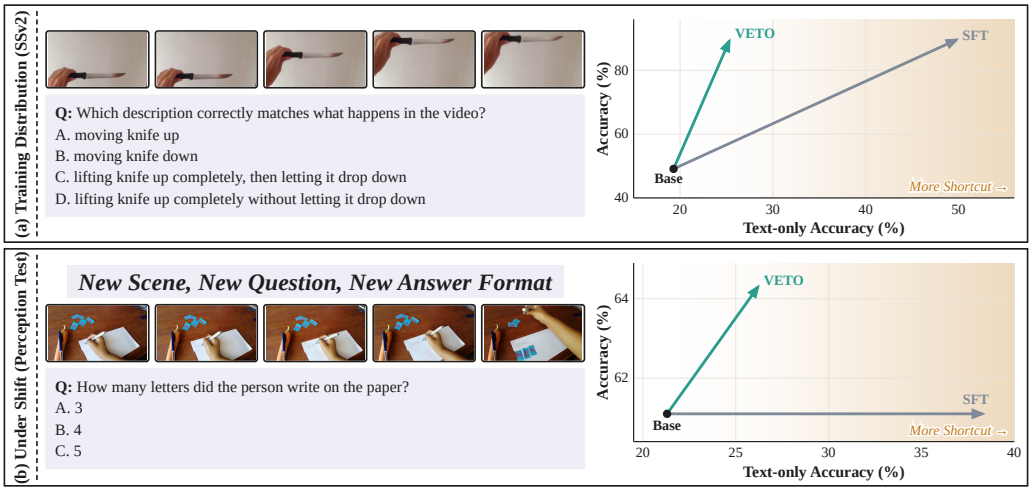

Figure 1: **SFT builds a shortcut, and VETO suppresses it.** Accuracy of Qwen2.5-VL-7B against its text-only accuracy, where a rise indicates the shortcut. (a) Within the training distribution (SSv2) SFT and VETO reach a similar gain in accuracy, but the shortcut grows much more under SFT. (b) Under the shift (Perception Test) VETO gains accuracy whereas SFT gains only the shortcut.

the gain of the fine-tuned model over the base model is limited, while its accuracy from the text alone still rises above the base model on every benchmark. In other words, the shortcut fine-tuning added persists where it no longer holds. SFT can thus appear to fine-tune the model successfully while silently teaching it to rely on spurious correlations rather than the video, leaving it vulnerable to any departure from its training distribution and diluting what makes it a VideoLLM. Filtering the data before fine-tuning does not prevent the shortcut, and the rise in accuracy hides it afterward, so the intervention must come during fine-tuning, where the model can be made to ground its answer in the full input rather than the shortcut alone.

This motivates our method, **Video-Evidence Training via Orthogonal projection** (VETO), which keeps the fine-tuning update out of the directions where the text alone is read. VETO first calibrates the subspace carrying the shortcut from the activations produced by the text-only input at every layer, once per backbone. It then projects the LoRA update onto the orthogonal complement of that subspace during fine-tuning, so that the update cannot enter the directions in which the shortcut would be learned. As VETO constrains only the update, it leaves the prior the base model already carries untouched and allows the model to adapt to the video rather than to the shortcut.

We evaluate VETO on the training distribution across seven VideoLLMs from five language models and on six unseen benchmarks. On the training distribution, VETO reaches nearly the same gain as SFT on every backbone, while the rise in text-only accuracy under SFT is largely suppressed. Under the shift VETO gains more over the base model than SFT, as Fig. 1(b) illustrates, and its reliance on the text rises far less than under SFT. VETO requires no definition of which regularity counts as the shortcut, no change to the data, loss, or schedule, and no cost beyond a projection per step. Across experiments, the shortcut emerges not as an inevitable prior that fine-tuning must keep for its gain but as what stands between a VideoLLM and the video it is given.

2 Shortcut Formation under Fine-Tuning

In this section we ask how the reliance of a VideoLLM on the text changes over fine-tuning under the standard recipe for adapting it to a target video QA task, SFT with LoRA (Hu et al., 2021). The base model is Qwen2.5-VL-7B (Bai et al., 2025b) and the target dataset is Something-Something v2 (SSv2) (Goyal et al., 2017a), whose answers come from a fixed set of action templates. Under distribution shift we evaluate on Perception Test (Patraucean et al., 2023), whose videos, questions, and options differ from SSv2 and whose options are written freely rather than from templates.

2.1 Blind Filtering and the Shortcut

Dataset preparation starts from the action template SSv2 assigns to every clip. We group the templates into families, where the templates of a family share the same objects and scene and differ only in what happens over time. Examples are direction (*moving X left* vs. *moving X right*) and state transition (*covering X* vs. *uncovering X*). The templates of a family are thus hard negatives of one another, and no other template can serve as one. Each clip becomes one *item*, a multiple-choice question in which the template of the clip is the correct option and the distractors are other templates rendered with the objects of the clip, with the families listed in Appx. A.

Blind filtering removes the items that the base model already answers from the text alone. The *full* input pairs the video with the question and options, and the *text-only* input is the full input without the video. We query the base model three times per item under the text-only input with the option order shuffled each time and remove the item when it answers correctly in all three, so that a lucky guess does not count. The filter applies to the training and evaluation splits alike, and on the evaluation split the base model reaches 19.3% text-only accuracy, below chance, as Appx. B details. The *shortcut* is therefore any rise in text-only accuracy over fine-tuning, a route to the answer that runs through the text alone and bypasses the video.

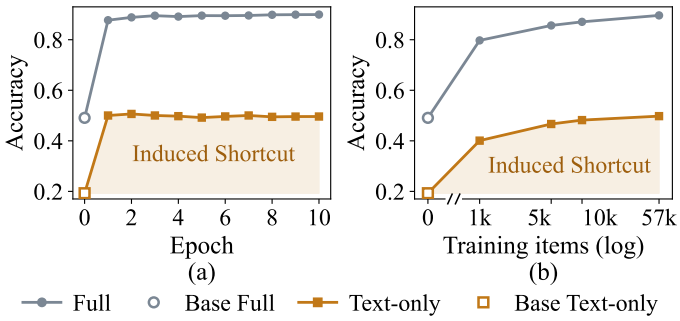

Figure 2: **The shortcut forms regardless of training epoch and data size.** (a) The shortcut is present from the first epoch and persists. (b) The shortcut is substantial at every training set size, already at 1k items. Mean over three seeds.

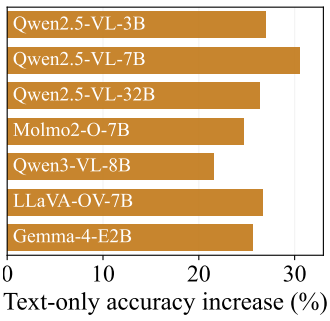

Figure 3: **The shortcut forms regardless of backbone.** Every VideoLLM gains large text-only accuracy after SFT.

2.2 Blind Filtering Does Not Prevent Shortcut Formation

Blind filtering removes every training item that the base model solves from the text alone, so SFT would seem to have nothing to learn from the text. Yet as Fig. 2(a) shows, the text-only accuracy rises together with the accuracy on the full input over the course of SFT, and the model ends up solving a substantial fraction of the evaluation set without the video. SFT thus builds a route through the text that the base model does not have.

The shortcut forms regardless of training epoch. Fig. 2(a) tracks text-only accuracy over epochs, and it is far above chance from the first epoch. The shortcut is thus not overfitting late in training but part of what SFT learns from the start, and early stopping cannot avoid it.

The shortcut forms regardless of data size. Fig. 2(b) plots the text-only accuracy against the training set size from 1k to all 57k items, and it is far above chance already at 1k and grows with the data as the accuracy does. The shortcut is thus not the noise of a small sample that more data would average out, but a regularity of the distribution that any sample of it carries.

The shortcut forms regardless of backbone. Fig. 3 gives the text-only gain for seven VideoLLMs from five language models and sizes up to 32B, and each answers at least a fifth more of the evaluation set from the text alone than its base model does. The shortcut is thus not particular to one language model or scale but what SFT does with the data.

These three observations suggest that what the shortcut rests on is present regardless of the training schedule, the data scale, or the backbone. Blind filtering leaves only the items that the base model does not solve from the text alone, but that does not mean their text carries no information about the answer. The options each item receives in the dataset preparation are distinct sentences, and any regularity between their wording and the correct answer is there for SFT to pick up even where the base model does not read it. The regularity thus persists in the data under every variation, and it becomes the shortcut once SFT learns to read it.

2.3 Why the Shortcut Matters

The purpose of SFT is to fit the target distribution for downstream tasks, and when that distribution carries regularity between the question and the answers, learning it can form a shortcut. From a Bayesian perspective this may not be a defect. A model that estimates a prior from training data and combines it with the visual evidence at inference does what the objective asks, and a rise in accuracy is consistent with that. Whether the shortcut is a problem therefore turns on two properties of what has been learned: what the shortcut reads and whether the visual evidence still enters the decision.

What the shortcut reads decides where it remains valid. The language prior the base model already carries reflects which actions tend to occur with which objects, and it holds where those facts hold. The shortcut is instead a regularity of the training set, and such a regularity may not hold once the questions and options are written differently. Tab. 1 tests this prediction on Perception Test, our evaluation set under shift, filtered as SSv2 is. On SSv2 the gap between the accuracy with the video and the accuracy from the text alone widens after fine-tuning, so the model gains more from the video than before and the shortcut sits alongside the reading of the video rather than replacing it. On Perception Test the same model reaches the accuracy of the base model it is fine-tuned from, while its accuracy from the text alone rises by 17.0%p, so the gap narrows and the shortcut has displaced much of what it drew from the video. What fine-tuning adds still raises the text-only accuracy on a format it was never trained on, yet there it replaces part of the video reading and adds nothing to accuracy. The shortcut is therefore a problem rather than a prior the model needs. What it reads holds only within the training distribution, and under the shift it displaces the visual evidence instead of adding to it. The remaining question is where to intervene.

Table 1: Accuracy with the full input and from text only, on SSv2 and Perception Test.

		Full	Text-only
SSv2	Base	49.1%	19.3%
	SFT	89.6%	49.8%
Perception Test	Base	61.1%	21.3%
	SFT	61.1%	38.3%

The training data are already blind-filtered, yet the shortcut forms through them. Curating them further would require naming the regularity, which Sec. 2.2 does not. Once training ends, the shortcut is hard to detect, which leaves little room for a remedy before or after training. Intervening on the training schedule is no easier, since the shortcut is in place regardless of epoch, data size, or backbone. What remains is training itself, and a constraint at that stage would not teach the model more than fine-tuning already does, but it can suppress the shortcut as it forms.

3 Video-Evidence Training via Orthogonal Projection

In this section we propose Video-Evidence Training via Orthogonal projection (VETO), which constrains the fine-tuning update to the orthogonal complement of the subspace the text alone occupies, as Fig. 4 outlines. We first identify the input directions that carry the shortcut (Sec. 3.1) and then keep the update out of those directions (Sec. 3.2).

3.1 Locating the Shortcut Subspace

We identify the subspace carrying shortcut information by collecting activations under text-only input, where the model receives the question and its options but no video. Let $\mathcal{D}_{\text{cal}} = \{q_i\}$ be a calibration set drawn from the same pool as the training data and spread evenly across the families. It is held out of training and used without labels, as only its activations under the text-only input enter the calibration. For each linear module m in layer l with weight $W^{l,m} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we collect the input activations, and for brevity we drop the module index m below.

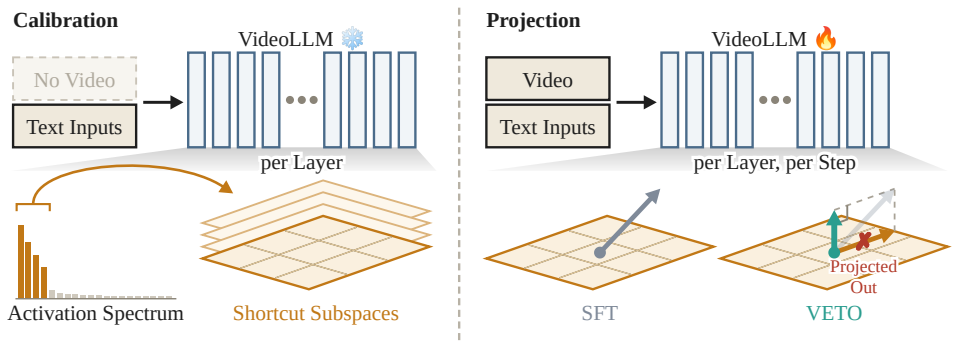

Figure 4: **Pipeline of VETO.** Calibration, once per VideoLLM, collects the activations of the text-only input at every layer and takes the leading directions of their spectrum as the shortcut subspace. During fine-tuning, the LoRA update is projected off that subspace after each optimizer step, so that the part along the shortcut directions is removed and the rest is kept, whereas SFT keeps both.

Text-only activations are collected by running a forward pass for each calibration item q_i under the text-only input and taking the input activations at every position. Concatenated over all items, they form the columns of $H^l_{\text{text}} \in \mathbb{R}^{d_{\text{in}} \times n_{\text{text}}}$, where n_{text} is the total number of positions over the calibration set rather than the number of items.

Shortcut subspace is defined from the second-moment matrix of the text-only activations,

$$C^l \;=\; \frac{1}{n_{\text{text}}} H^l_{\text{text}} (H^l_{\text{text}})^\top. \tag{1}$$

The matrix is left uncentered so that the mean direction of the text-only activations stays inside the subspace, since the text alone occupies that direction as much as any other. We eigendecompose $C^l = U^l \Lambda^l (U^l)^\top$ with eigenvalues in descending order, and retain the smallest rank r^l whose leading eigenvalues sum to at least a fraction ρ of the trace,

$$r^l \;=\; \min \Big\{ r : \textstyle\sum_{j \le r} \lambda^l_j \;\ge\; \rho \cdot \operatorname{tr}(C^l) \Big\}, \tag{2}$$

forming $U^l_1 \in \mathbb{R}^{d_{\text{in}} \times r^l}$, whose columns span the shortcut subspace at layer l.

The text-only activations of items outside the calibration set fall inside U^l_1 to nearly the fraction ρ imposed on the calibration items, and independent draws of the set span nearly the same subspace. The subspace is thus a property of the backbone rather than of the particular items it is estimated from, and a few hundred items suffice to estimate it, as Appx. C shows. The one quantity ρ sets is the retained rank r^l and thereby how much of the input space the update cannot enter.

3.2 Projecting Out the Shortcut

Given the shortcut subspace identified, we keep the fine-tuning update ΔW^l out of it by projecting the update onto the orthogonal complement of the subspace, $\Delta W^l \leftarrow \Delta W^l P^l$, with the projector

$$P^l \;=\; I - U^l_1 (U^l_1)^\top. \tag{3}$$

We instantiate the update with LoRA (Hu et al., 2021), which parameterizes it as a low-rank product $\Delta W^l = B^l A^l$, so that the projection acts on the input factor. At each optimizer step, the accumulated update is projected on its input side as $\Delta W^l P^l$ by right-multiplying the down-projection factor, $A^l \leftarrow A^l P^l$. The optimizer states are left as they are, since the projection is idempotent and returns the component of A^l inside the subspace to zero, which Appx. D verifies over training. This yields the invariance

$$\big(W^l + \Delta W^l P^l\big)\,\mathbf{u} \;=\; W^l \mathbf{u} \qquad \forall\, \mathbf{u} \in \operatorname{span}(U^l_1), \tag{4}$$

which guarantees that the response of the fine-tuned model to any input direction inside the shortcut subspace remains that of the base model.

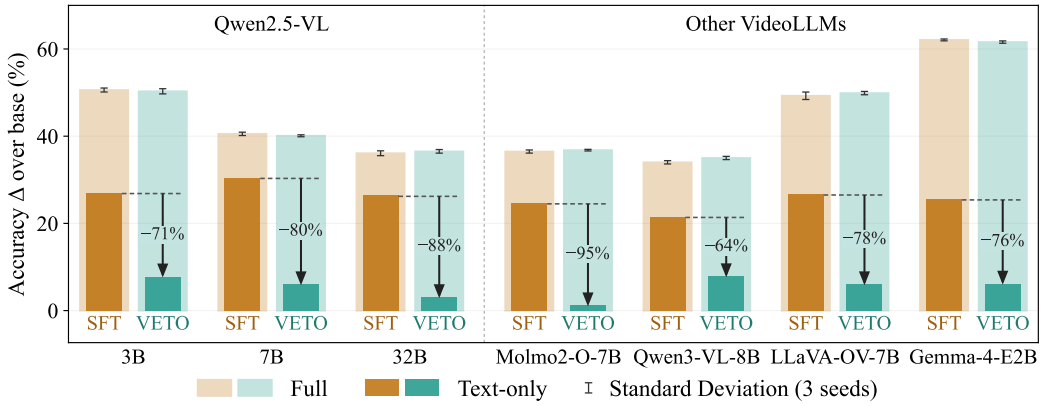

Figure 5: **VETO removes a substantial part of the shortcut across VideoLLMs.** The full-input gain is nearly the same as under SFT, whereas the text-only gain is suppressed as intended.

Two properties follow from Eq. 4. First, the constraint is preventive rather than corrective, since the shortcut directions are unreachable from the first gradient step, so no pathway forms along them. Second, the constraint acts on the update alone. What the base model already reads from the text as a prior or a valid deduction stays as it is by construction, and only the text-only pathway that fine-tuning would add is suppressed. The success criterion is therefore the removal of the gap induced by fine-tuning, $\text{acc}^{\text{SFT}}_{\text{text-only}} - \text{acc}^{\text{base}}_{\text{text-only}}$, not the base prior itself.

Both properties concern the directions inside the span of U_1^l. Within the span the removal is exact, as $\Delta W^l P^l U_1^l$ is identically zero, and in practice the ratio $\|\Delta W^l U_1^l\|_F / \|\Delta W^l\|_F$ stays at numerical zero in every run. The span may not cover the whole shortcut, since the subspace is estimated from calibration activations, and any component outside it remains open to the update. The constraint also says nothing about where the displaced learning settles, and within the span it keeps out every new use of the text, the valid ones along with the shortcut. What VETO removes, where fine-tuning routes it, and whether the gain of fine-tuning survives the constraint are the questions for experiments.

4 Experiments

4.1 Setup

Fine-tuning baseline is standard SFT with rank-16 LoRA on the 57k items of the blind-filtered SSv2 training set of Sec. 2.1, with 16 frames per video, for up to 10 epochs at learning rate 10^{-4}. We select checkpoints from a held-out subset of the evaluation split that no reported accuracy uses, the same for every run. VETO trains under the identical recipe with the projection of Sec. 3.2 at $\rho = 0.95$. Every run uses a single NVIDIA B200 GPU, and Appx. D lists the remaining hyperparameters, the baseline implementations, and the seed statistics.

Backbones span seven VideoLLMs from five language models, Qwen2.5-VL (Bai et al., 2025b) at 3B, 7B, and 32B, Qwen3-VL-8B (Bai et al., 2025a), LLaVA-OneVision-7B (Li et al., 2025), Molmo2-O-7B (Clark et al., 2026), and Gemma4-E2B (Team et al., 2026), each fine-tuned with the recipe above and calibrated against its own activations.

Evaluation covers the SSv2 evaluation split and six benchmarks the fine-tuning never sees, Perception Test (Patraucean et al., 2023), Vinoground (Zhang et al., 2024), AoTBench (Xue et al., 2025), MVBench (Li et al., 2024), TVBench (Cores et al., 2024), and TempCompass (Liu et al., 2024). We blind-filter each set as in Sec. 2.1, and accuracy uses the highest-logit option letter as the answer. Appx. B gives the counts.

4.2 Fine-Tuning Gains Without the Shortcut

We measure accuracy changes relative to the base model across model and backbone scales. Fig. 5 reports the gain with the full input and from the text alone, and the shortcut reduction

$1 - \Delta_t^{\text{VETO}}/\Delta_t^{\text{SFT}}$, with Δ_t the text-only gain over the base model. On every VideoLLM, VETO suppresses the shortcut formation, while SFT and VETO gain alike with the full input. SFT raises text-only accuracy along with full-input accuracy, whereas VETO raises it far less. The shortcut that rests on the regularities of the training set may offer the VideoLLM a ground for its answers on the training distribution, but it is not indispensable. The VideoLLM trained with VETO reaches nearly the same gain without it, so VETO suppresses no prior that accuracy requires.

Table 2: Accuracy (Acc.) and text-only gain over the base model (Text Δ) of Qwen2.5-VL-7B on the blind-filtered subset of Perception Test, by area. Best in **bold**, second best underlined.

	Physics		Semantics		Abstraction		Memory		All	
	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓
Base	56.9	–	78.0	–	52.4	–	41.5	–	61.1	–
SFT	59.0±2.0	+17.4±1.8	75.0±1.0	+17.7±2.9	52.2±0.6	+17.5±2.0	**45.0**±1.5	+6.2±4.2	61.1±0.7	+17.0±1.1
VETO	**63.0**±1.1	**+6.7**±3.8	**78.5**±0.6	**+4.0**±1.6	**54.3**±1.2	**+3.9**±0.7	**45.0**±0.7	**+1.2**±2.8	**64.3**±0.7	**+4.9**±2.2

Table 3: Accuracy (Acc.) and text-only gain over the base model (Text Δ) of Qwen2.5-VL-7B on the blind-filtered subsets of five further benchmarks, as in Tab. 2.

	Vinoground		AoTBench		MVBench		TVBench		TempCompass	
	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓	Acc.↑	Text Δ ↓
Base	55.9	–	41.3	–	**58.0**	–	42.4	–	64.0	–
SFT	63.5±0.4	+13.5±1.7	53.1±0.2	+10.9±1.3	56.5±0.8	+7.8±0.5	44.3±0.8	+7.6±0.4	65.8±0.6	+4.0±0.6
VETO	**66.9**±2.0	**+7.7**±0.8	**55.2**±1.3	**+5.6**±2.5	**58.0**±1.0	**+3.5**±0.3	**46.4**±0.4	**+3.0**±1.5	**67.6**±0.3	**+2.4**±0.7

4.3 Shortcut under Distribution Shift

Tab. 2 reports the accuracy on Perception Test, whose videos, questions, and options all differ from SSv2. SFT raises text-only accuracy by 17.0%p overall and more than VETO in every area, so the shortcut of SFT carries over to this benchmark. On the full input, however, SFT is no better than the base model, whereas VETO improves over the base model while raising text-only accuracy much less. On this benchmark SFT keeps the shortcut but loses its gain, while VETO suppresses the shortcut and keeps its gain. Tab. 3 extends the comparison to five further unseen benchmarks, and VETO exceeds SFT on every one of them. VETO also brings far less of the shortcut into every benchmark than SFT, so the suppression carries over as well. Suppressing the shortcut is thus what makes the VideoLLM gain more than SFT under the shift.

4.4 Where the VideoLLM Grounds Its Reasoning

As VETO suppresses the shortcut while largely preserving the accuracy of the VideoLLM, a pathway other than the shortcut has to carry that accuracy. This leads us to investigate where the VideoLLM grounds its reasoning on Qwen2.5-VL-7B throughout this section. To this end, we compare SFT and VETO on the full input and text alone using two probes.

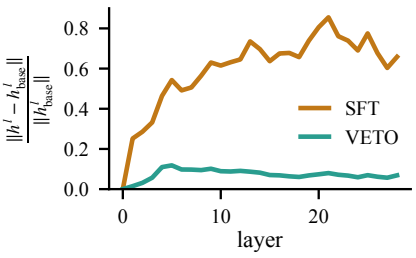

Figure 6: Relative deviation from the base model per layer, text-only input.

The first probe asks where fine-tuning changes the computation on text. Under the text-only input we measure how far fine-tuning has moved the representation at each layer. The measure is the norm of the difference between the hidden states of the last token in the fine-tuned and the base model, relative to the norm in the base model. As presented in Fig. 6, SFT moves the representation further with depth, so the change that fine-tuning makes to the computation on text accumulates over the layers. VETO keeps it close to the base model at every layer. The invariance of Eq. 4 thus carries through the network.

The second probe asks how confident the VideoLLM is in its answer, so we take the entropy over the options as a share of its maximum. The entropy is maximal when the options are equally likely and zero when one option takes all the probability, so low entropy means the VideoLLM has settled on an answer. Tab. 4 reports the entropy under the text-only and full input. Their difference is the part of the uncertainty that the video resolves and appears as Video Δ. SFT already settles on an answer without the video, so little is left for the video to resolve. VETO stays unsettled without the video, and the video resolves most of that uncertainty. The confidence therefore comes from the video under VETO and from the text under SFT.

Table 4: Entropy over the options as % of its maximum. Lower % for more confidence.

		Text-only	Full	Video Δ
SSv2	SFT	16%	2%	14%
	VETO	68%	2%	66%
Perception Test	SFT	33%	14%	19%
	VETO	81%	18%	63%

The shortcut could be taken for a prior the VideoLLM needs to reason since SFT raises the accuracy on the training distribution, but the two probes say otherwise. The first shows that computation on text alone stays close to the base model at every layer, while VideoLLM reaches accuracy with the video, so the accuracy does not rest on the prior. The second shows that the confidence SFT gains without the video is not a prior awaiting evidence but an answer already settled, and SFT carries this settled answer under the shift, where the regularity behind it no longer holds. With VETO, the confidence appears only once the video is present, so the accuracy the VideoLLM reaches with the video is carried by its reading of the video, on the training distribution and under the shift.

4.5 Other Interventions and Ablations

Table 5: Alternatives to VETO. Text Δ is the text-only gain over the base model, lower for more suppression.

	Method	Acc.	Text Δ
	SFT	89.6%	+30.5%p
Data	Reshuffled options	89.4%	+30.2%p
Loss	LMH (Clark et al., 2019)	83.4%	+13.3%p
	mDPO (Wang et al., 2024)	87.0%	+26.1%p
Update	Random subspace	89.6%	+30.4%p
	VETO	89.2%	+6.0%p

Table 6: Ablations of the constraint with the same columns as in Tab. 5.

	Setting	Acc.	Text Δ
	SFT	89.6%	+30.5%p
ρ	0.90	88.6%	+12.9%p
	0.95	89.2%	+6.0%p
	0.99	89.5%	+1.1%p
Project	At init. only	88.7%	+30.0%p
	After training	82.4%	+2.4%p
	Every step	89.2%	+6.0%p

As the shortcut forms regardless of epoch, data size, or backbone, an intervention can act at the data, the loss, or the update. Tab. 5 compares VETO with alternative interventions on Qwen2.5-VL-7B, with each row reporting the accuracy and the text-only gain.

Intervening at the *data* requires knowing which regularity to remove. Reshuffling the answer position every epoch removes the one regularity evident without inspection and leaves the shortcut almost unchanged. With the question fixed, the regularity that remains lies in the text of the options, and a rule that removes it is written for one dataset and does not transfer to the next.

Intervening at the *loss* does not require defining the regularity. Learned-Mixin +H (LMH) (Clark et al., 2019) reduces the gradient where the text alone gives the answer, and multimodal direct preference optimization (mDPO) (Wang et al., 2024) penalizes an answer that stays as likely without the video. LMH suppresses part of the shortcut at a high accuracy cost, and mDPO builds nearly the same shortcut at lower accuracy. Adjusting the loss thus trades accuracy for suppression, whereas VETO keeps the update out of the shortcut and preserves the accuracy.

Intervening at the *update* reaches where the shortcut is written. A random subspace of the same rank leaves the accuracy and text-only accuracy of SFT almost unchanged, so the rank removed is not what matters. What matters is the directions the calibration locates, and removing those directions alone is how VETO keeps the accuracy and suppresses the shortcut with no regularity to define.

Tab. 6 varies the constraint itself. A larger ρ closes more of the input space and suppresses more of the shortcut, and the stronger suppression does not cost accuracy. When the projection is applied is important as well, since projected once at initialization the update drifts back into the subspace and the shortcut forms as under SFT. Projecting once after training suppresses the shortcut at the cost

of accuracy, as it discards what was learned along those directions. The shortcut thus accumulates over the steps in between, and only a projection at every step keeps the update out of the subspace throughout. This is what makes VETO preventive rather than corrective.

5 Related Work

Shortcuts and language priors have drawn extensive analysis since vision-language models (VLMs) came to rest on LLMs. In the image domain VLMs answer from the question alone (Goyal et al., 2017b; Agrawal et al., 2018), and in video many questions are likewise answered from text alone (Yang et al., 2025). The video is read as what typically happens rather than what the clip shows (Buch et al., 2022; Lei et al., 2023; Zhang et al., 2024; Liu et al., 2024). Fine-tuning then adds what its data teach, as a VLM reproduces what is frequent in its instruction data regardless of the visual input (Li et al., 2023; Chen & Li, 2026). Such an added regularity is the shortcut, as distinct from the prior of the base model, and it predicts the label while it may not hold under distribution shift (Geirhos et al., 2020; Dancette et al., 2021; Agrawal et al., 2018). Remedies remove the bias from the data (Chen et al., 2020; Shvetsova et al., 2025), down-weight it in the loss through a question-only model (Cadene et al., 2019; Clark et al., 2019; Niu et al., 2021) or a text-only penalty against the base model (Chen & Li, 2026), or correct the trained model through preference optimization (Wang et al., 2024) or its decoding through contrastive decoding (Yang et al., 2025). Each first defines the shortcut as a quantity and then removes or penalizes it, but a shortcut outside the definition lies outside the remedy (Dancette et al., 2021). Unlike for images, remedies for video still concentrate on the data and decoding rather than on training itself (Yang et al., 2025). VETO instead constrains the update, keeping it out of the directions in which the shortcut would be learned without defining it, so that the shortcut is suppressed as it forms rather than defined and removed.

Constraining the update of fine-tuning to a subspace separates what a model maintains from what it modifies. Continual learning keeps gradients in the null space of features from earlier tasks (Wang et al., 2021) and the adapter subspaces of successive tasks orthogonal (Wang et al., 2023) or interference-free (Liang & Li, 2024), and model editing projects an edit onto the null space of the keys of knowledge it preserves (Fang et al., 2025). The same projection has been applied to representations rather than to updates (Ravfogel et al., 2020; Belrose et al., 2023), including to remove a text-like subspace from the visual tokens of a VLM (Saini et al., 2026). In all of these the subspace is defined by what the model already has and specified for it. LoRA (Hu et al., 2021) gives the constraint its natural object, since its update enters the weights as a few new directions (Shuttleworth et al., 2025). VETO then defines the subspace by what fine-tuning would read from the text, so that the same projection prevents rather than preserves. VETO adds only a projection per step and needs no definition of the shortcut, which turns a constraint built to preserve into one that prevents.

6 Conclusion

This paper addresses the shortcut SFT builds in a VideoLLM from textual regularities of its training data. The shortcut forms across VideoLLMs, yet it is not what fine-tuning requires for its gain and even costs accuracy under distribution shift. VETO suppresses the shortcut as it forms, preserves the gain of fine-tuning on the training distribution, and gains more than SFT under shift. What VETO calibrates is not a list of regularities but where the text alone is read, so the same projection can make a model use all of its input in other modalities. More broadly, what fine-tuning adds to a VideoLLM is set not only by its data but by the directions its update is allowed to enter, and constraining the update is a means of intervention the data, the loss, and the trained model do not offer.

7 Limitations and Future Directions

Our analysis of the shortcut rests on multiple-choice video QA, where text-only accuracy can be measured directly. VETO itself is not restricted to this format, but free-form answering has no direct measure of the shortcut yet and is left to future work. The projection of VETO is instantiated with LoRA, and its extension to other adapters on the input side of the update is a further direction. The main experiments fine-tune on SSv2 alone, and the same suppression of the shortcut is observed when fine-tuning on two further datasets, as Appx. F reports.

AI USE STATEMENT

In this work, we used generative AI tools for correcting grammar, formatting tables, and placing figures in the paper. We have not used generative AI tools for preprocessing the training data, which follows the template of each dataset, and all videos and question-answer prompts come from publicly available datasets, with datasets such as Perception Test used under their license for academic research. The formulas in the paper were written by the authors, and generative AI tools were used only to typeset them in LaTeX. Additionally, we used generative AI tools to search for prior work that may overlap with the proposed method, so as to identify if there are any existing works with the same idea. We have reviewed all AI-assisted work. We have checked that every referenced work in the paper is relevant to the point the authors make with it. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Video-Evidence Training via Orthogonal projection (VETO) is a training-time technique that constrains how a pretrained VideoLLM is fine-tuned. It operates on existing models and leaves the prior of the base model untouched, so it inherits their capabilities and limitations, including the biases in their pretraining data, and does not correct them. VETO is not a safety filter and does not certify that a model reasons from the video. A model fine-tuned with VETO should therefore not be taken as faithful to its visual input beyond what is measured.

Our experiments rely on publicly available VideoLLMs and on public video datasets used under their licenses, and no new data involving human subjects were collected. This work is intended for academic research. Any application of methods that shape what a model learns during fine-tuning should document the calibration data and constraints used and follow the model and dataset licenses.

REPRODUCIBILITY STATEMENT

We will make the source code for VETO publicly available, including the calibration, projection, blind filtering, training, and evaluation pipelines together with the data manifests that define every split. The appendix provides the implementation details, including the fine-tuning recipe and hyperparameters, the construction of the calibration set and of the evaluation splits, and the pretrained models and datasets used in our evaluation, and every reported value is a mean over the number of seeds stated there.

REFERENCES

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4971–4980. IEEE, 2018.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. LEACE: Perfect linear concept erasure in closed form. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=awIpKpwTwF.

Shyamal Buch, Cristóbal Eyzaguirre, Adrien Gaidon, Jiajun Wu, Li Fei-Fei, and Juan Carlos Niebles. Revisiting the "video" in video-language understanding. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2907–2917. IEEE, 2022.

Remi Cadene, Corentin Dancette, Hedi Ben-younes, Matthieu Cord, and Devi Parikh. RUBi: Reducing unimodal biases for visual question answering. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. Counterfactual samples synthesizing for robust visual question answering. In *CVPR*, 2020.

Yangneng Chen and Jing Li. Language bias in LVLMs: From in-depth analysis to simple and effective mitigation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=0tdnyH7uXp.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language Processing (EMNLP-IJCNLP)*, pp. 4069–4082, 2019.

Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Mohammadreza Salehi, Rohun Tripathi, Sangho Lee, Zhongzheng Ren, Chris Dongjoo Kim, Yinuo Yang, Vincent Shao, Yue Yang, Weikai Huang, Ziqi Gao, Taira Anderson, Jianrui Zhang, Jitesh Jain, George Stoica, Winson Han, Ali Farhadi, and Ranjay Krishna. Molmo2: Open weights and data for vision-language models with video understanding and grounding, 2026. URL https://arxiv.org/abs/2601.10611.

Daniel Cores, Michael Dorkenwald, Manuel Mucientes, Cees GM Snoek, and Yuki M Asano. Lost in time: A new temporal benchmark for videollms. *arXiv preprint arXiv:2410.07752*, 2024.

Corentin Dancette, Remi Cadene, Damien Teney, and Matthieu Cord. Beyond question-based biases: Assessing multimodal shortcut learning in visual question answering. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1554–1563. IEEE, 2021.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations*, volume 2025, pp. 16366–16396, 2025.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, 2023.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pp. 5842–5850, 2017a.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017b.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jie Lei, Tamara Berg, and Mohit Bansal. Revealing single frame bias for video-and-language learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 487–507, 2023.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13872–13882. IEEE, 2024.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-onevision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=zKv8qULV6n.

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.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 292–305, 2023.

Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647. IEEE, 2024.

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.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. Counterfactual vqa: A cause-effect look at language bias. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12695–12705. IEEE, 2021.

nostalgebraist. interpreting GPT: the logit lens, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. LessWrong blog post.

Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens Continente, Larisa Markeeva, Dylan Sunil Banarse, Skanda Koppula, Joseph Heyward, Mateusz Malinowski, Yi Yang, Carl Doersch, Tatiana Matejovicova, Yury Sulsky, Antoine Miech, Alexandre Fréchette, Hanna Klimczak, Raphael Koster, Junlin Zhang, Stephanie Winkler, Yusuf Aytar, Simon Osindero, Dima Damen, Andrew Zisserman, and Joao Carreira. Perception test: A diagnostic benchmark for multimodal video models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=HYEGXFnPoq.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7237–7256, 2020.

Harshvardhan Saini, Samyak Jha, Yiming Tang, and Dianbo Liu. When language overwrites vision: Over-alignment and geometric debiasing in vision-language models. *arXiv preprint arXiv:2605.08245*, 2026.

Reece S Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. LoRA vs full fine-tuning: An illusion of equivalence. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=xp7B8rkh7L.

Nina Shvetsova, Arsha Nagrani, Bernt Schiele, Hilde Kuehne, and Christian Rupprecht. Unbiasing through textual descriptions: Mitigating representation bias in video benchmarks. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29050–29059. IEEE, 2025.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Aditya Chawla, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, Johan Ferret, Filippo Galgani, Sertan Girgin, Jean-Bastien Grill, Maarten Grootendorst, Jiaxian Guo, Cassidy Hardin, Yanzhang He, Steven M. Hernandez, Omri Homburger, Léonard Hussenot, Juyeong Ji, Armand Joulin, Aishwarya Kamath, Parnian Kassraie, Olivier Lacombe, Preethi Lahoti, Gaël Liu, Gus Martins, Luciano Martins, Tatiana Matejovicova, Ramona Merhej, Nikola Momchev, Sneha Mondal, Ryan Mullins, Sindhu Raghuram Panyam, Shreya Pathak, Sarah Perrin, André Susano Pinto, Etienne Pot, Angéline Pouget, Alexandre Ramé, Sabela Ramos, Douglas Reid, David Rim, Morgane Rivière, Karsten Roth, Louis Rouillard, Omar Sanseviero, Pier Giuseppe Sessa, Shane Settle, Danila Sinopalnikov, Sara Smoot, Piotr Stanczyk, Andreas Steiner, Lawrence Stewart, Ilya Tolstikhin, Michael Tschannen, Anton Tsitsulin, Nino Vieillard, Renjie Wu, Pingmei Xu, Haichuan Yang, Edouard Yvinec, Biao Zhang, Li Zhang, Joe Zou, Nicolas Aagnes, Abdelrahman Abdelhamed, Jakub Adamek, Shivani Agrawal, Shubham Agrawal, Ibrahim Alabdulmohsin, Jean Baptiste Alayrac, Uri Alon, Chandramouli Amarnath, Ankesh Anand, Chrysovalantis Anastasiou, Setareh Ariafar, François-Xavier Aubet, Kyriakos Axiotis, Federico Barbero, Joelle Barral, Alexei Bendebury, Urs Bergmann, Stanley Bileschi, Kat Black, Mathieu Blondel, Sebastian Borgeaud, Arthur Bražinskas, Ryan Burnell, Robert Busa-Fekete, Mu Cai, Daniele Calandriello, Glenn Cameron, Charlotte Caucheteux, Rahma Chaabouni, Garima Chadha, Jetha Chan, Blake Jianhang Chen, Jesse Chen, Lin Chen, Xu Chen, Derek Cheng, Tzu hsiang Chien, Nikolai Chinaev, Yi Chou, Zhaohui Chu, Benjamin Coleman, Pooja Consul, Sam Conway-Rahman, Scott Crowell, Dylan Cutler, Vivek Dani, Samira Daruki, Anil Das, Daniel Deutsch, Nishanth Dikkala, Li Ding, Qiuhan Ding, Shenil Dodhia, Konstantin Donhauser, Tulsee Doshi, Anca Dragan, Alex Druinsky, Sahil Dua, Zoltan Egyed, Danielle Eisenbud, Daniel Eppens, Cindy Fan, Bahare Fatemi, Yassir Fathullah, Vlad Feinberg, Milen Ferev, Sebastian Flennerhag, Takumi Fujimoto, João Gabriel Oliveira, Isaac Galatzer-Levy, João Gante, Simon Geisler, Soham Ghosal, Antonious M. Girgis, Tamara von Glehn, Alec Go, Alhaad Gokhale, Alex Grills, Yiming Gu, Mayank Gupta, Pramod Gupta, Guru Guruganesh, Raia Hadsell, Hamza Harkous, Jitendra Harlalka, Demis Hassabis, Anja Hauth, Joe Heyward, Arian Hosseini, Chih-Yang Hsia, I-Hung Hsu, Xiaopeng Huang, Yangsibo Huang, Kevin Hui, Adrian Hutter, Te I, Fotis Iliopoulos, Advait Jain, Ganesh Jawahar, Ziwei Ji, Qilin Jin, Melvin Johnson, Kandarp Joshi, Arun Kandoor, Wang-Cheng Kang, Koray Kavukcuoglu, Mehran Kazemi, Kathleen Kenealy, Amr Khalifa, Phoebe Kirk, Ivan Korotkov, Suraj Kothawade, Vitaly Kovalev, Neel Kovelamudi, Adam Kraft, Ravin Kumar, Vivek Kumar, Harish Kuppam, Justin Lannin, Chen-Yu Lee, Seungji Lee, Dmitry Lepikhin, Alon Levkovitch, Dongdong Li, Qiujia Li, Valentin Liévin, Ethan Lin, Ziqian Lin, Casper Liu, Tianlin Liu, Tianqi Liu, Xin Liu, Ivan Lobov, Mayank Lunayach, Min Ma, Gagan Madan, Andrii Maksai, Eric Malmi, Michal Matuszak, Daniel McDuff, Gaurav Menghani, Maciej Mikuła, Daniil Mirylenka, Karolis Misiunas, Vedant Misra, Andreea Mitran, Kareem Mohamed, Maksim Mukha, Eric Noland, James O'Donnell, Brendan O'Donoghue, Kate Olszewska, Bernett Orlando, Wanqiong Pan, Rina Panigrahy, Unnati Parekh, Nicolas Perez-Nieves, Chunjong Park, Eric Paskie, Liqian Peng, Bryce Petrini, Slav Petrov, Jonas Pfeiffer, Bilal Piot, Martyna Plomecka, Siim Poder, Octavio Ponce, Arijit Pramanik, David Racz, Anish Rajan, Michelle Ramanovich, Anand Rao, Marvin Ritter, Vitor Rodrigues, Evan Rosen, Mikołaj Rybiński, Noveen Sachdeva, Michaël E. Sander, Rohit Sathyanarayana, Sagar Savla, Samuel Schmidgall, Tal Schuster, George Scrivener, Benoit Seguin, Andrew Sellergren, Aliaksei Severyn, Izhak Shafran, Dhruv Shah, Bobak Shahriari, Yuan Shangguan, Ashish Shenoy, Pradeep Shenoy, Rakesh Shivanna, Pauline Sho, Lucas Spangher, Wojciech Stokowiec, Tim Strother, Yao Su, Yinghao Sun, Mukund Sundararajan, Andrea Tacchetti, Mor Hazan Taege, Pouya Tafti, Jean Tarbouriech, Chetan Tekur, Shantanu Thakoor, Rahul Thapa, Madeleine Traverse, Lenart Treven, Tao Tu, Chien Te Tung, Çağlar Ünlü, Petar Veličković, Malini Pooni Venkat, Sagar Gubbi Venkatesh, Vidya Venkiteswaran, Francesco Visin, Alex Vitvit-

skyi, Kiran Vodrahalli, Weiyi Wang, Xin Wang, Tris Warkentin, Jan Wassenberg, John Wieting, Cindy Wu, Lechao Xiao, Hao Xu, Yuhui Xu, Fuzhao Xue, Arun Yadav, Jun Yan, Antoine Yang, Lin Yang, Ming-Hsuan Yang, Ziyu Ying, Jae Hyeon Yoo, Morteza Zadimoghaddam, Sajjad Zafar, Fred Zhang, Jiageng Zhang, Jianyi Zhang, Xiaofan Zhang, Chao Zhao, David Zhou, and Chen Zou. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Fei Wang, Wenxuan Zhou, James Y Huang, Nan Xu, Sheng Zhang, Hoifung Poon, and Muhao Chen. mdpo: Conditional preference optimization for multimodal large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8078–8088, 2024.

Qi Wang, Yanrui Yu, Ye Yuan, Rui Mao, and Tianfei Zhou. VideoRFT: Incentivizing video reasoning capability in MLLMs via reinforced fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=3pORFyKzh1.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 184–193. IEEE, 2021.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671, 2023.

Ziyang Wang, Jaehong Yoon, Shoubin Yu, Md Mohaiminul Islam, Gedas Bertasius, and Mohit Bansal. Video-rts: Rethinking reinforcement learning and test-time scaling for efficient and enhanced video reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 28126–28140, 2025b.

Bo Wu, Shoubin Yu, Zhenfang Chen, Joshua B. Tenenbaum, and Chuang Gan. STAR: A benchmark for situated reasoning in real-world videos. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=EfgNF5-ZAjM.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9772–9781. IEEE, 2021.

Zihui Xue, Mi Luo, and Kristen Grauman. Seeing the arrow of time in large multimodal models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=OYciB30Z4n.

Yiming Yang, Yangyang Guo, Hui Lu, and Yan Wang. Vidlbeval: Benchmarking and mitigating language bias in video-involved lvlms. *arXiv preprint arXiv:2502.16602*, 2025.

Jianrui Zhang, Mu Cai, and Yong Jae Lee. Vinoground: Scrutinizing lmms over dense temporal reasoning with short videos. *arXiv preprint arXiv:2410.02763*, 2024.

APPENDIX

A SSV2 FAMILY CONSTRUCTION

Of the 174 SSv2 action templates (Goyal et al., 2017a), 47 come in hard-negative families of two to four templates. The templates of a family share the same object placeholders and, once rendered for a given clip, differ only in temporal content. We use these 47 in 16 families. Families fall into three tiers by the kind of temporal evidence they require. Direction families (horizontal, vertical, relative, two camera-motion families, object-relative) require the direction of motion. State-transition families (containment, cover, open-close) require the order of two states. Outcome families (lift, fall, spin, slant, poke, push, plug) require the result of an action that is visible only at the end of the clip.

Each clip yields one multiple-choice question. The correct option is the template of the clip, and the distractors are the sibling templates re-rendered with the placeholders of that clip. For the eight two-template families we pad to four options with designated near-family distractors, which serve as options and never as answer sources, and the three-template family keeps its three options. Table A lists every family with its templates. The retained templates cover roughly 38% of SSv2 clips.

Table A: The SSv2 families with their number of templates, items in the blind-filtered training pool and evaluation split, and templates in order of frequency. [X] marks the object placeholders of SSv2.

Family	Templ.	Train	Eval	Templates
Horizontal	4	8,562	756	Pushing [X] from left to right / Pushing [X] from right to left / Pulling [X] from right to left / Pulling [X] from left to right
Vertical	4	7,786	789	Moving [X] up / Moving [X] down / Lifting [X] up completely without letting it drop down / Lifting [X] up completely, then letting it drop down
Relative motion	4	3,507	483	Moving [X] and [X] away from each other / Moving [X] and [X] closer to each other / Moving [X] and [X] so they pass each other / Moving [X] and [X] so they collide with each other
Camera, horizontal	4	3,495	419	Moving away from [X] with your camera / Turning the camera left while filming [X] / Turning the camera right while filming [X] / Approaching [X] with your camera
Camera, vertical	4	3,136	266	Moving [X] away from the camera / Turning the camera downwards while filming [X] / Turning the camera upwards while filming [X] / Moving [X] toward the camera
Object-relative	2	1,753	383	Moving [X] away from [X] / Moving [X] closer to [X]
Containment	4	5,438	874	Putting [X] into [X] / Taking [X] out of [X] / Stuffing [X] into [X] / Pulling [X] out of [X]
Cover	2	4,761	656	Covering [X] with [X] / Uncovering [X]
Open–close	2	2,053	480	Closing [X] / Opening [X]
Lift	2	2,829	170	Lifting up one end of [X] without letting it drop down / Lifting up one end of [X], then letting it drop down
Fall	2	2,446	680	[X] falling like a rock / [X] falling like a feather or paper
Spin	2	1,765	452	Spinning [X] that quickly stops spinning / Spinning [X] so it continues spinning
Slant	2	638	88	Putting [X] that can't roll onto a slanted surface, so it slides down / Putting [X] that can't roll onto a slanted surface, so it stays where it is
Poke	4	3,511	378	Poking [X] so lightly that it doesn't or almost doesn't move / Poking [X] so it slightly moves / Poking [X] so that it falls over / Poking [X] so that it spins around
Push	3	3,879	618	Pushing [X] so that it slightly moves / Pushing [X] so that it falls off the table / Pushing [X] so that it almost falls off but doesn't
Plug	2	1,363	524	Plugging [X] into [X] / Plugging [X] into [X] but pulling it right out as you remove your hand
Total	47	56,922	8,016	

B BLIND FILTERING DETAILS

We run the base model on each split under the text-only input three times with the option order shuffled, and discard items it answers correctly in all three. The rule keeps an item whenever the base model fails at least once in three, and the base model answers 19.3% of the kept items from the text, below chance. The filter removes 10.4% of the training pool, 63,526 to 56,922, and 10.7% of the evaluation pool, 8,977 to 8,016. The same filter applied to the benchmarks of Sec. 4.1 keeps 5,637 of the 7,392 multiple-choice items of the released training split of Perception Test (Patraucean et al., 2023), 665 of the 1,000 of Vinoground (Zhang et al., 2024), 1,350 of the 1,757 valid items of AoTBench (Xue et al., 2025), 2,727 of the 3,586 of MVBench (Li et al., 2024), 1,767 of the 2,205 of TVBench (Cores et al., 2024), and 1,097 of the 1,580 of TempCompass (Liu et al., 2024). Every result under the shift is reported on these subsets.

Table B: Calibration size sweep, Qwen2.5-VL-7B, three seeds per row, mean ± sd over seeds. Random rows draw n items from the training pool as nested prefixes of a shuffled order, and stratified rows draw $n/16$ items per family, which is how the calibration set of the paper is drawn. Coverage is measured on $1{,}000$ held-out items, overlap is averaged over the three seed pairs, and the random baseline for both is $r^l/d_{\text{in}} \approx 0.07$.

Draw	n	Coverage	Overlap between seeds	Rank $\bar{r}$
random	10	0.8694 ± 0.0051	0.63	103
random	50	0.9205 ± 0.0062	0.78	171
random	100	0.9299 ± 0.0071	0.83	195
random	200	0.9387 ± 0.0024	0.88	217
random	1,000	0.9471 ± 0.0002	0.95	246
random	2,000	0.9484 ± 0.0003	0.97	251
random	4,000	0.9487 ± 0.0001	0.98	252
stratified	200	0.9444 ± 0.0003	0.90	241
stratified (used)	**400**	$\mathbf{0.9485 \pm 0.0005}$	0.93	253
stratified	500	0.9485 ± 0.0005	0.94	251
stratified	600	0.9496 ± 0.0001	0.94	257
stratified	800	0.9499 ± 0.0004	0.95	256
stratified	1,000	0.9504 ± 0.0005	0.96	258

Throughout the paper, chance refers to uniform guessing over the options, 0.257 on the SSv2 evaluation split since one family has three options. The answer letters of the split are nearly balanced, so guessing by their empirical frequencies gives 0.257 and always choosing the most frequent letter gives 0.266. On the blind-filtered SSv2 evaluation split the base model reaches 0.193 under the text-only input, below all three. On every filtered set the text-only accuracy of the base model lies below uniform chance by construction of the filter, so Text Δ is measured against the base model rather than against chance. Under VETO the text-only accuracy on the SSv2 evaluation split returns to chance, 0.253 against 0.257, whereas under SFT it rises to twice chance.

The blind filter uses the base model only, so it takes the language prior of the base model out of the measurement and nothing else. Any text-only accuracy above that of the base model after fine-tuning is therefore added by fine-tuning, and this addition is the shortcut of Sec. 2.1.

C Calibration Set Size

The shortcut subspace of Sec. 3.1 is estimated from a calibration set, and this section asks how many items the estimate needs, how much it depends on which items are drawn, and how the energy fraction ρ sets its rank. Coverage plateaus at 400 stratified items, and $\rho = 0.95$ is the last value at which the subspace stays small, and the paper uses these two. We measure *coverage*, the fraction of text-only activation energy of $1{,}000$ held-out items that falls inside U_1^l, that is, $\mathrm{tr}\big((U_1^l)^\top C_{\text{held}}^l U_1^l\big)/\mathrm{tr}(C_{\text{held}}^l)$ averaged over all adapted modules. A subspace that generalizes perfectly would reach the $\rho = 0.95$ imposed on the calibration items, and a random subspace of the same rank would reach $r^l/d_{\text{in}} \approx 0.07$. Between two subspaces estimated from different draws we measure *overlap*, $\|(U_a^l)^\top U_b^l\|_F^2/\min(r_a^l, r_b^l)$, which equals 1 when one subspace contains the other and again r^l/d_{in} for two random subspaces.

Tab. B sweeps the calibration size from 10 to $4{,}000$ randomly drawn items for Qwen2.5-VL-7B with three seeds, where each seed shuffles the training pool once and the sets are nested prefixes of that order, so that the comparison across sizes is not confounded by which items are drawn. Coverage rises quickly and then saturates, with 50 items already at 0.9205, $1{,}000$ items at 0.947, and the next $3{,}000$ items adding 0.002, and the standard deviation over seeds stays below 0.01 at every size. The estimate is thus a property of the backbone rather than of the items.

The lower half of Tab. B repeats the sweep with draws stratified over the 16 families, which is how the calibration set of the paper is drawn, and stratification lifts the whole curve, as 200 stratified items reach 0.9444 where 200 random items reach 0.9387. Along the stratified curve the gain from 200 to 400 items is 0.004, whereas the gain from 400 to $1{,}000$ items is 0.002 in total, and the three seeds at

any size differ by at most 0.0005, so the curve plateaus at 400 items. We take the calibration size at the onset of this plateau, 400 items, which sit within 0.002 of the ρ imposed on the calibration items, and the rank retained there is within five dimensions of the rank at $1{,}000$ items. Overlap between seeds keeps rising past the plateau, from 0.93 at 400 items to 0.96 at $1{,}000$, and since coverage does not rise with it, the items beyond the plateau settle which of the near-degenerate tail directions of the calibration spectrum enter U_1^l rather than how much shortcut energy it captures.

The energy fraction ρ sets the rank retained by Eq. 2, and Tab. C follows it across the spectrum of the stratified calibration set of 400 items, with $d_{\text{in}} = 3{,}584$ and the $\rho = 0.95$ row the projector used throughout. Coverage on held-out items tracks ρ to within a point at every rank, so the estimate generalizes equally well whether it retains six directions or two thousand, and ρ is a choice of how much of the input space to close. The rank grows slowly up to $\rho = 0.95$, where 95% of the held-out energy is captured within 7% of the dimensions, and steeply beyond it, with $\rho = 0.99$ retaining a quarter of the input space for four more points of coverage. We therefore use 0.95, the last point at which the subspace is still small. Training under 0.90, 0.95, and 0.99 in Tab. 6 shows what the rank buys, as a larger ρ suppresses more of the shortcut while the accuracy stays within seed variation.

Table C: Retained rank and held-out coverage against ρ.

ρ	Rank $\bar{r}$	$\bar{r}/d_{\text{in}}$	Coverage
0.5	6	0.2%	0.544
0.7	20	0.6%	0.719
0.8	39	1.1%	0.810
0.9	106	3.0%	0.903
0.95	**250**	**7.0%**	**0.948**
0.99	928	25.9%	0.985
0.999	2,211	61.7%	0.997

Table D: The seven VideoLLMs, their language model family and parameter count, effective parameters for Gemma4-E2B, and the accuracy (%) of the base model on the SSv2 evaluation split.

VideoLLM	Language model	Params	Full	Text-only
Qwen2.5-VL-3B	Qwen2.5	3B	38.5	22.4
Qwen2.5-VL-7B	Qwen2.5	7B	49.1	19.3
Qwen2.5-VL-32B	Qwen2.5	32B	53.1	24.9
Qwen3-VL-8B	Qwen3	8B	51.5	30.0
LLaVA-OneVision-7B	Qwen2	7B	39.2	24.8
Molmo2-O-7B	Olmo	7B	53.5	22.6
Gemma4-E2B	Gemma 4	2.3B	25.8	25.7

D Implementation Details

Data splits follow Sec. 2.1. The blind-filtered SSv2 training set holds 56,922 items, of which 400 are drawn by stratified sampling over the families as the calibration set and held out, leaving 56,522 for training. The blind-filtered evaluation split holds 8,016 items, of which 200 drawn at random serve to select the checkpoint and the remaining 7,816 are the test set on which every accuracy of Sec. 4 is reported. The calibration, training, selection, and test sets are therefore pairwise disjoint.

Backbones of Sec. 4.1, Qwen2.5-VL (Bai et al., 2025b), Qwen3-VL (Bai et al., 2025a), LLaVA-OneVision (Li et al., 2025), Molmo2 (Clark et al., 2026), and Gemma 4 (Team et al., 2026), are listed in Tab. D with their language model family and size, and with the accuracy of the base model on the SSv2 evaluation split with the full input and the text-only input.

Training uses LoRA adapters (Hu et al., 2021) on the attention and MLP projections of every decoder layer, with the down projection left untouched, and Tab. E lists the settings. We optimize with AdamW (Loshchilov & Hutter, 2017) under a cosine schedule for up to 10 epochs with early stopping on the selection accuracy, where the Qwen2.5-VL-7B runs of SFT and VETO complete all 10 epochs for the epoch curve of Fig. 2(a), with the checkpoint selected as in every other run. Attention uses FlashAttention-2 (Dao, 2024) where the backbone supports it. Qwen2.5-VL-7B is trained with three seeds and reported as mean and standard deviation, the other backbones use the same three seeds, and Tab. H gives the seed count of every row of Tab. 5 and Tab. 6. SFT and VETO share all of the above, and VETO adds only the projection of Sec. 3.2 after every optimizer step.

Table E: Hyperparameters shared by SFT and VETO.

LoRA rank / scaling / dropout	16 / 32 / 0.05
Adapted modules	q, k, v, o, gate, up
Optimizer	AdamW, lr 10^{-4}, weight decay 0.01
Schedule	cosine, 10% warm-up, clip 1.0
Items per update	8
Precision	bf16
Epochs / patience	up to 10 / 3 on the selection set
Frames / resolution	16 / $\leq 360 \times 420$
Micro-batch	4 (Qwen, all ablation rows), 1 (others)
Selection / test set	200 / 7,816 items of the evaluation split
Seeds	3 per configuration unless stated
Calibration	400 prompts, text-only, $\rho = 0.95$
Hardware	one NVIDIA B200 per run

Calibration and projection run once before training. For each adapted module the input activations at every text position of the 400 calibration prompts under the text-only input are collected from the base model, their uncentered second moment is decomposed, and the leading directions that carry $\rho = 0.95$ of the energy form U_1^l, which for Qwen2.5-VL-7B has rank 250 on average over the 168 adapted modules. The projection $A \leftarrow A\,(I - U_1^l U_1^{l\top})$ is applied to the A factor of each adapter after every optimizer step. The two projection ablations of Tab. 6 differ only in when this projection runs. At initialization only, the adapter is projected once and then trains without the projection, so the leakage grows from zero as the update re-enters the subspace. After training, the adapter of the SFT run is projected once. The AdamW moments are not projected in any run. What they accumulate along U_1^l is removed from the weights at the next step, and the leakage $\|A^l U_1^l\|_F / \|A^l\|_F$ measured at every epoch stays at numerical zero in every VETO run, so the accumulation never reaches the weights. Since the projection is applied after every step, it is equivalent to discarding the component of each update inside U_1^l, so the constraint can equally be read as a hard regularizer on the update that sets its component along the shortcut directions to zero rather than penalizing it. Under the initialization-only ablation the same ratio rises from zero to 0.20 after the first epoch and to 0.30 by the fourth in each of the three seeds. Under SFT without any projection it stands at 0.34 at the selected checkpoint in each of the three seeds, above the 0.26 that a random direction of the same rank would give, so the update of SFT leans toward the text directions.

Loss baselines in Sec. 4.5 follow the published objectives with the answer letter as the prediction target. Learned-Mixin +H (Clark et al., 2019) needs a bias model that carries the text-only pathway, for which we fine-tune the same backbone with the same LoRA on the text-only prompt and cache its letter logits z_b for every training item, so that the bias model enters the loss as a fixed signal. The main model then trains on $z = \log \mathrm{softmax}(z_m) + g(h)\, \log \mathrm{softmax}(z_b)$, where z_m are its own letter logits, h its hidden state at the answer position, and $g(h) = \mathrm{softplus}(w^\top h + b)$ a learned gate. The input h of the gate is normalized to unit RMS and the gate is clamped at 5. The loss is $\mathrm{CE}(z, y) + \lambda\,(-H(\mathrm{softmax}(g(h)\, \log \mathrm{softmax}(z_b))))$ at $\lambda = 0.36$, the value published for VQA-CP, and the model predicts from z_m alone at evaluation. The LMH row trains under the budget of the other rows, up to 10 epochs with early stopping on the selection set. mDPO (Wang et al., 2024) is applied as published to the instruction-tuned base model, which also serves as the reference policy, at learning rate 10^{-5} on the training items. The preferred response is the correct letter and the rejected one a wrong letter drawn per item, and the rejected video keeps a random fifth or less of the frames and blacks out the rest, following the crop of the original. The loss is the published sum of the preference, conditional, and anchored terms at $\beta = 0.1$ and $\delta = 0$.

Decoding-time method. Visual contrastive decoding (VCD) (Leng et al., 2024) contrasts the letter logits under the original frames with those under noised frames at inference, and we implement it for our backbone with $\alpha = 1$, $\beta = 0.1$, and $T = 500$ as published. VCD is undefined without frames, so it leaves the text-only accuracy, and with it the shortcut, as it is. Tab. F applies it to SFT and VETO on the SSv2 evaluation split, and neither the accuracy nor the shortcut moves under it.

Table F: Accuracy under VCD, three seeds.

	Full	VCD
SFT	89.6$\pm$0.4	89.7$\pm$0.4
VETO	89.2$\pm$0.2	89.2$\pm$0.2

Table G: Training cost on one B200 for one epoch of the 57k items, mean ± standard deviation over repeated measurements.

Method	One-off (GPU-h)	Per update (s)	One epoch (GPU-h)	×SFT
SFT	–	2.24±0.01	4.40±0.02	1.00
VETO	0.03±0.00	2.36±0.02	4.67±0.05	1.06
LMH	17.77±0.33	2.26±0.03	22.21±0.34	5.05
mDPO	5.28±0.38	4.44±0.01	13.98±0.39	3.18

Training cost. Tab. G measures the cost of each row of Tab. 5 under the main recipe. The one-off column is the calibration of VETO, the bias model and its cached logits of LMH, with the bias model extrapolated from a measured segment to its five epochs, and the cached reference log-probabilities of mDPO. Reshuffling costs as SFT, and the random subspace and the ablations cost as VETO. The per-update time is averaged over five runs and the one-off cost over three. VETO trains at nearly the cost of SFT, whereas LMH and mDPO cost several times more.

Table H: Accuracy and text-only accuracy (%) on the SSv2 evaluation split for the base model and the rows of Tab. 5 and Tab. 6, Qwen2.5-VL-7B, mean ± standard deviation over n seeds.

	n	Accuracy	Text-only accuracy
Base	–	49.1	19.3
SFT	3	89.6±0.4	49.8±0.8
Reshuffled options	3	89.4±0.2	49.4±1.0
LMH	1	83.4	32.6
mDPO	1	87.0	45.3
Random subspace	3	89.6±0.0	49.6±2.0
VETO ($\rho = 0.95$)	3	89.2±0.2	25.3±0.6
$\rho = 0.90$	3	88.6±0.8	32.1±3.9
$\rho = 0.99$	3	89.5±0.1	20.4±0.4
Projected at init. only	3	88.7±0.3	49.2±1.2
Projected after training	3	82.4±0.4	21.6±0.1

Seed statistics for Sec. 4.5. Tab. H lists, for the base model and every row of Tab. 5 and Tab. 6, the mean and standard deviation over seeds of the accuracy and the text-only accuracy on the SSv2 evaluation split, from which Text Δ is the mean text-only accuracy minus that of the base model.

E The Two Pathways in Detail

Where the video enters the text positions. The subspace U_1^l is estimated from text-position activations under the text-only input, and the constraint keeps the update out of it alone. Whether the video can still reach the update therefore depends on where the text-position activations fall once video tokens are present. We measure, on the base model and the calibration items, the fraction of text-position activation energy that lies inside U_1^l, that is, $\mathrm{tr}((U_1^l)^\top C^l U_1^l)/\mathrm{tr}(C^l)$ with C^l the uncentered second moment of the activations at the positions after the last vision token, under the text-only input and under the original video. Under the text-only input the fraction equals the ρ imposed by construction. Under the original video it falls, as Tab. I shows, most at the output projection of attention, which is where the attention output carries what was read from the video tokens. The video thus moves part of the text-position activations outside the calibrated subspace, and this part is where the update remains free, which is the route through which VETO learns from the video. The same table gives the fraction under the text-only input on held-out SSv2 items and on the text of the six benchmarks under shift. On held-out SSv2 it stays at the imposed value, and on the six benchmarks, whose text shares nothing with SSv2, it is close to four fifths, against about one fourteenth for a random subspace of the same rank, so the subspace estimated on the text of SSv2 covers most of where the text alone is read on other text as well. The calibration collects these activations without video tokens in front of the text, so the text tokens sit at the positions of the text-only input. The subspace nevertheless holds close to nine tenths of the text-position energy once the video tokens are present, as the table shows, so the shift of positions between the two conditions does not move the

Table I: Fraction of text-position activation energy inside U_1^l on the base Qwen2.5-VL-7B, averaged over modules and by module type, under the text-only input on the calibration items, on held-out SSv2, and on the six benchmarks under shift, and under the original video on the calibration items.

Input	Mean	q	k	v	o	gate	up
Text only, calibration items	0.950	0.950	0.950	0.950	0.950	0.950	0.950
Text only, held-out SSv2	0.947	0.948	0.948	0.948	0.945	0.946	0.946
Text only, Perception Test	0.806	0.818	0.818	0.818	0.789	0.797	0.797
Text only, Vinoground	0.823	0.832	0.832	0.832	0.807	0.818	0.818
Text only, AoTBench	0.778	0.791	0.791	0.791	0.756	0.771	0.771
Text only, MVBench	0.808	0.820	0.820	0.820	0.790	0.800	0.800
Text only, TVBench	0.786	0.800	0.800	0.800	0.757	0.781	0.781
Text only, TempCompass	0.839	0.850	0.850	0.850	0.826	0.830	0.830
Original video, calibration items	0.879	0.891	0.891	0.891	0.851	0.876	0.876

text activations out of it. A fifth of the energy of the text-only activations of the six benchmarks lies outside the subspace, yet the text-only gain on them stays well below that of SFT, so the directions the shortcut uses lie mostly inside the calibrated subspace even for text the calibration never saw.

Entropy over the options. The entropy reported in Sec. 4.4 is that of the distribution the VideoLLM places over the option letters at the answer position, $-\sum_o p(o) \log p(o)$ in nats, averaged over the evaluation items, so that it runs from zero, where one option takes all the probability, to $\ln n$ for n options, where the options are equally likely. The share of the maximum in Tab. 4 divides the entropy of each item by the $\ln n$ of its own options. The last column is the difference between the first two, that is, the part of the entropy without the video that the video resolves. Tab. J gives the values in nats, averaged over three seeds on 2,000 items of each set.

Table J: Entropy over the options in nats, mean $\pm$ sd over three seeds, on 2,000 items of the SSv2 evaluation split and of Perception Test, Qwen2.5-VL-7B.

	SSv2		Perception Test	
	Text-only	Full	Text-only	Full
Maximum ($\ln 4$ or $\ln 3$)	1.39	1.39	1.10	1.10
Base	1.04	0.84	0.90	0.63
SFT	$0.22_{\pm 0.08}$	$0.03_{\pm 0.01}$	$0.37_{\pm 0.09}$	$0.16_{\pm 0.05}$
VETO	$0.93_{\pm 0.08}$	$0.03_{\pm 0.02}$	$0.89_{\pm 0.02}$	$0.20_{\pm 0.07}$

The answer through the layers. Reading the option logits from the hidden state of every layer through the final norm and the output head (nostalgebraist, 2020) gives the accuracy each layer would reach on its own. Under the text-only input the base model and VETO stay at chance at every layer, whereas SFT rises above chance from the last third of the layers and reaches its text-only accuracy at the end, so the text-only pathway forms in the same late layers where Fig. 6 shows the representation drifting most. With the video, SFT and VETO rise at the same layers to nearly the same accuracy, so the constraint does not change where the answer with the video is formed. Tab. K gives the accuracy at a quarter, a half, three quarters, and the full depth.

Table K: Accuracy read from the hidden state at four depths, under the text-only input and with the original video, on 2,000 items of the SSv2 evaluation split, Qwen2.5-VL-7B, mean over three seeds.

Input	Model	$L/4$	$L/2$	$3L/4$	L
Text only	Base	0.26	0.27	0.23	0.20
	SFT	0.26	0.26	0.40	0.50
	VETO	0.27	0.27	0.27	0.25
Original video	Base	0.25	0.26	0.29	0.49
	SFT	0.25	0.26	0.52	0.89
	VETO	0.27	0.26	0.47	0.89

Table L: Accuracy (%) with the full input, with the attention to the video blocked, and from text-only, on SSv2 and Perception Test, Qwen2.5-VL-7B, mean ± sd over three seeds.

	SSv2			Perception Test		
	Full	Blocked	Text-only	Full	Blocked	Text-only
Base	49.1	19.5	19.3	61.1	21.5	21.3
SFT	89.6±0.4	49.5±1.1	49.8±0.8	61.1±0.7	38.4±0.9	38.3±1.1
Random subspace	89.6±0.0	49.8±1.8	49.6±2.0	62.1±0.9	35.8±0.5	35.2±0.6
VETO	89.2±0.2	30.2±1.2	25.3±0.6	64.3±0.7	25.9±3.1	26.2±2.2
Share of correct answers kept under the block						
Base		29.7			25.2	
SFT		51.2±1.1			45.7±2.7	
VETO		30.2±1.1			26.3±4.1	

The video still enters the decision. Blocking the attention from text tokens to video tokens at inference (Geva et al., 2023) leaves the input intact and removes only the reading of it, where the text-only input omits the video itself, and Tab. L reports both. On SSv2 SFT loses most of its accuracy under the block, so the video enters its decision, and what remains is its accuracy from the text alone, so the shortcut keeps answering when the video cannot be read. The base model retains far less under the block, and this gap is what fine-tuning added to the shortcut. On Perception Test the block costs the base model more than SFT, so the video there carries evidence and SFT reads less of it than the base model, while what SFT retains is again its accuracy from the text alone. The shortcut therefore sits beside the reading of the video on the training distribution and in place of part of it under the shift, and in neither case is the video path closed.

Reliance on the video. The last rows of Tab. L give the share of the answers each model gets right with the full input that it gets right under the block. In every cell the accuracy under the block stays close to the accuracy from text alone, within a point under SFT, the base model, and the random subspace and within five points under VETO, so the block removes the reading of the video nearly as cleanly as removing the video itself. A random subspace of the same rank at every module leaves the accuracy under the block under the shift at the level of SFT and the accuracy within a point of it, inside the seed spread, so the gain VETO makes there rests on the directions its calibration locates rather than on the rank it removes.

F Fine-Tuning on Other Datasets

The main experiments fine-tune on SSv2 alone. This section reports fine-tuning on two further datasets, NExT-QA (Xiao et al., 2021) and STAR (Wu et al., 2021), whose questions and options are written freely rather than from templates. Tab. M reports fine-tuning Qwen2.5-VL-7B under the recipe of Sec. 4.1, on the blind-filtered training split of NExT-QA and a 20k subsample of that of STAR stratified over its question types, with the blind filter and calibration applied as to SSv2 in the main experiments. On both datasets the text-only accuracy rises far above VETO under SFT, whereas the gain on the full input is nearly the same under SFT and VETO, as across the backbones of Fig. 5.

Table M: Fine-tuning on NExT-QA and STAR, three seeds.

Dataset	Model	Acc.	Text Δ
NExT-QA	Base	70.9	–
	SFT	71.7±0.4	+8.1±0.7
	VETO	71.6±0.2	+1.3±0.2
STAR	Base	67.9	–
	SFT	70.5±0.2	+11.6±1.1
	VETO	70.9±0.6	+1.3±0.4