HOLOGRAM: A GRAPH-CONDITIONED CONTEXT STEERING FRAMEWORK FOR MULTILINGUAL, MULTIMODAL, AND MULTI-GRANULAR TOXIC CONTENT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, we introduce a novel graph-conditioned steering framework HOLOGRAM, which is among the first content moderating unified architecture that adapts a frozen VLM to these three axes of moderation heterogeneity through the same mechanism. For each post, HOLOGRAM constructs a semantic moderation neighborhood from training. Subsequently, a graph-based-encoder models how nearby examples support, duplicate, or contrast with one another. Its representations select representative artifacts to generate a query-conditioned low-rank context vector inside every frozen decoder layer of the VLMs. The resulting framework supports multiple modalities, languages, and supports labels at both, binary and fine-grained levels. We evaluate our framework across an array of established vector (and memory)-steering methods and prove that our graph-conditioned steering framework not only outperforms prior methods, but also yields high consistency. The results demonstrate the potential of local relational residual context addition as a general interface for heterogeneous social-media moderation and proves its strong usability, even towards production deployment. ***Warning: It may contain potentially toxic contents that could come from the dataset.***

1 INTRODUCTION

Toxic content moderation has garnered significant research interest in recent years (Hee et al., 2024b; Kapil & Ekbal, 2024; Rizwan et al., 2025b). Its scope has progressively broadened across diverse languages and modalities, encompassing everything from textual posts to multimodal hateful memes. The modern landscape of social media is inherently fractured across multiple axes of diversity, which renders the standard and often monolithic moderation approaches ineffective. A year back, META also released a documentation[1] featuring MARK ZUCKERBERG, detailing the complex landscape of toxic content moderation. These complexities in such contents primarily manifests in three distinct dimensions (Rizwan et al., 2025b). First, *multimodality*: toxic behavior frequently transcends pure text and embeds itself within a complex multimodal interaction setup; such as hateful memes, where the juxtaposition of seemingly benign images and text creates malicious intent. Second, *multilinguality that leads to cultural nuance*: toxic content proliferates not only in high-resource languages like ENGLISH but also in low-resource and code-switched settings (e.g., HINDI, AMHARIC), heavily relying on culturally specific idioms, implicit dog whistles, and rapidly evolving slangs. Third, *label granularity*: the definition of toxicity is not monolith; it spans a broad spectrum from simple binary classification (for example: `toxic` vs. `normal`) to other fine-grained categorizations encompassing hate-speech, harassment, stereotyping, and micro-aggressions among many others. Consequently, existing moderation systems often suffer from acute fragmentation typically requiring distinct, specialized models or exhaustive domain-specific fine-tuning in parameter-space for each language, modality, or taxonomic definition. Developing a singular, unified framework capable of generalizing across this highly ***heterogeneous*** landscape without catastrophic performance degradation remains a formidable open challenge, which we successfully attempt to solve in this work.

HOLOGRAM: To effectively navigate the aforementioned heterogeneity, we propose HOLOGRAM,

[1] https://about.fb.com/news/2025/01/meta-more-speech-fewer-mistakes/

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.

a novel graph-encoder based few-shot context steering mechanism. Unlike traditional in-context learning (ICL) (Rizwan et al., 2026) which relies heavily on expanding the context window that often lead to suboptimal context utilization, or existing activation steering methods that employ a singular memory block across the entire network (Wang et al., 2026), HOLOGRAM (refer to Sections 3 for problem formulation) takes a fundamentally different approach. It first constructs a similarity-ranked graph of few-shot examples to capture intricate, cross-modal, and cross-lingual contextual relationships. This structured context is then injected into the Vision-Language Model (VLM) via per-layer low-rank matrices. Acting as fine-grained contextual memory adapters, these matrices steer the model's intermediate representations at every individual layer. By dynamically adapting the VLM's latent space, HOLOGRAM seamlessly handles heterogeneity.

Key contributions and results

☞ **Novel context steering architecture**: We introduce HOLOGRAM, a unified framework for toxic content moderation. By leveraging a graph-encoder paired with per-layer low-rank matrices, our approach provides precise, layer-wise activation steering, overcoming the bottleneck of single-block memory methods like PMT (Wang et al., 2026).
☞ **Handling moderation heterogeneity**: We demonstrate that HOLOGRAM effectively bridges the heterogeneity gap in social media content. It provides a single, robust mechanism capable of seamlessly processing multimodal inputs, multilingual text, and multi-granular label definitions without requiring task-specific fine-tuning in parameter-space or excessively long prompts as employed in ICL.
☞ **Substantial improvement in performance**: Extensive evaluations across 36 distinct (model, dataset) configurations reveal that HOLOGRAM consistently surpasses both standard prompting baselines (zero-shot, chain-of-thoughts, ICL) and recent training-based steering methods (PMT, LIVE). It exhibits particularly profound improvements in non-ENGLISH settings (e.g., AMHARIC and HINDI), yielding absolute performance gains of up to 14% in some configurations.
☞ **Efficiency and scaling behavior**: Our analysis reveals that HOLOGRAM enables highly efficient context utilization, with 4-shot contextual steering generally providing the optimal context-performance trade-off. Furthermore, HOLOGRAM significantly enhances parameter efficiency, empowering smaller 2B parameter VLMs to achieve performance at par with their 8B counterparts, thereby paving a compute-efficient path for future moderation systems.

2 RELATED WORK

Toxic content detection: Research in content moderation has expanded from foundational ENGLISH datasets like HATEXPLAIN (Mathew et al., 2021) to multilingual corpora like HASOC (Mandl et al., 2019) (HINDI) and Ayele et al. (2023) (AMHARIC), revealing that multilinguality fundamentally shifts label spaces and social contexts beyond mere translation. This complexity amplifies in hateful memes, where FHM (Kiela et al., 2020) demands joint visual-textual reasoning, and BHM (Hossain et al., 2024) extends this to CODE-MIXED BENGALI settings. Consequently, VLMs are extensively investigated in this domain (Rizwan et al., 2025a; Hee et al., 2024b; Jha et al., 2024; Cao et al., 2023; Hee et al., 2024a; Rizwan et al., 2026; Lu et al., 2025; Adak et al., 2025), while datasets like TOXICTAGS (Swain et al., 2026) and MULTI3HATE (Bui et al., 2025) further underscore the deep cultural dependencies of toxic content.
Relational adaptation of frozen models: Early approaches like TASK VECTORS (Ilharco et al., 2023) composed task-specific behaviors by interpolating weight residuals. Subsequent works explored activation space manipulations (Todd et al., 2024; Hendel et al., 2023), injecting hidden states as static steering vectors to modulate generation; a concept generalized by representation engineering (Zou et al., 2025). To overcome static vector limitations, PREFIX MEMORY-TUNING (PMT) (Wang et al., 2026) stabilizes context modulation by decoupling learnable prefixes from attention and utilizing an external memory module. Specifically for VLMs, the LEARNABLE IN-CONTEXT VECTOR (LIVE) framework (Peng et al., 2024) distills multimodal demonstrations into a single parameter-efficient vector. Recently, (Bhaskar et al., 2026) proposed steering vectors for hateful memes alignment.

3 PROBLEM FORMULATION

Formally, we define a moderation regime as a tuple $\tau = (m, \ell, \mathcal{Y})$, comprising the modality m, language ℓ, and a target policy label set $\mathcal{Y}$. An input instance $x_q = (t_q, I_q)$ consists of a textual

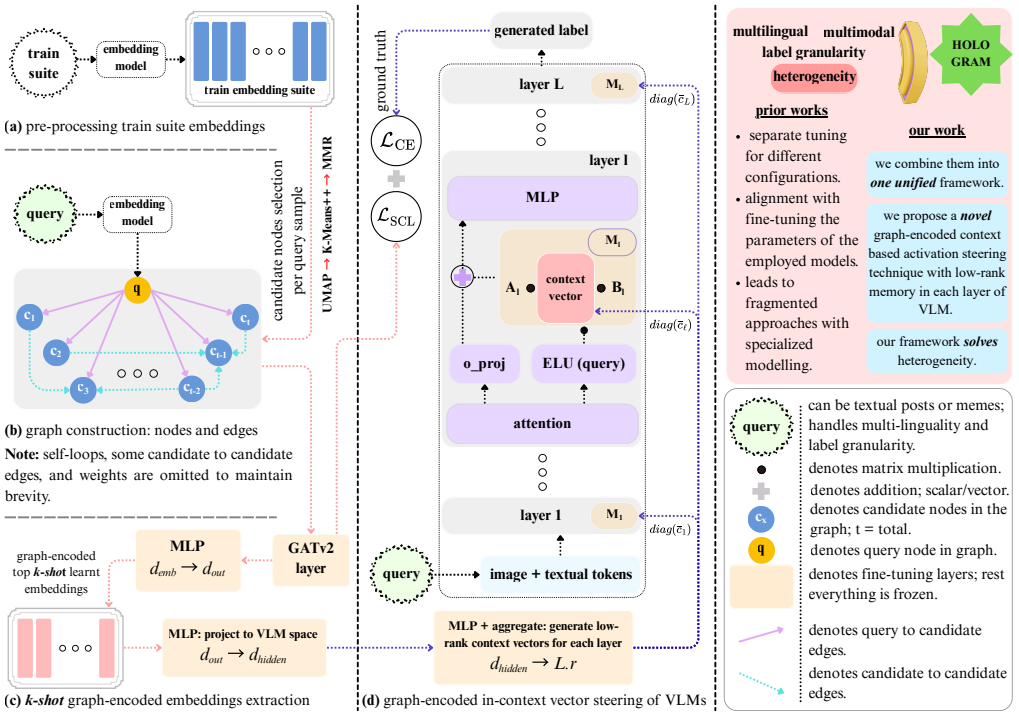


Figure 1: Overview of HOLOGRAM: Parts **(a)** to **(d)** picturize the different steps of our unified steering framework. A brief contrast of prior works with ours is also presented on top-right. Details are covered in Sections 3 and 4.

utterance t_q and an optional image I_q. Given a labeled training bank $\mathcal{B}$ and a frozen auto-regressive VLM, our goal is to adapt the VLM for toxic content moderation to the target regime τ. Crucially, all learnable parameters are strictly confined to an external graph encoder and a context builder interface; the VLM and upstream embedding models remain entirely frozen (refer Section 4). To achieve this unified formulation, the proposed architecture must satisfy three strict forms of invariance: **(i) label-space invariance:** Adapting to a new label set $\mathcal{Y}$ should not require a new training paradigm. Instead, the method must adapt to different target regime τ, constraining the frozen VLM to generate exactly one member from label set $\mathcal{Y}$ based on the input prompt, **(ii) modality invariance:** The architecture must robustly handle multimodal inputs (i.e., textual posts/memes in our case) without necessitating structural changes to the decoder or to the adaptation interface. For this reason, we use a VLM to handle both modalities, and **(iii) linguistic invariance:** The identical graph-to-context builder framework must be seamlessly reused across different languages. Ultimately, a single shared architecture handles all regimes, relying purely on the regime-specific evidence supplied by the labeled bank $\mathcal{B}$ for the tuple τ and the policy description to govern the VLM's behavior.

4 HOLOGRAM

This section walks through the entire HOLOGRAM pipeline step by step. In order to make the explanation concrete in the upcoming subsections, we sometimes use running examples. Note that, as inspired by prior works in low-resource steering of generative models (refer to Section 2 for related works), we propose a novel activation-steering technique powered by graph-supplied few-shot context. Below we discuss each component of our framework and also present it in Figure 1.

4.1 OVERVIEW

HOLOGRAM adapts a frozen VLM for toxic content detection through dynamic context steering, circumventing the need for task-specific manipulations or model's parameter updates. We formulate

the task as purely generative (e.g., to output "toxic" or "normal") and employ strictly zero-shot input prompt in the token space, completely omitting few-shot demonstrations from the input prompt to align with the other methods like PMT (Wang et al., 2026), and LIVE (Peng et al., 2024). Instead, relevant support examples are aggregated via a GATv2 graph-encoder and routed to the VLM exclusively through an internal context steer in the activation space. These graph representations dynamically generate a low-rank context vector that is injected into each frozen decoder layer of the VLM as a residual state after the self-attention block. This structural decoupling ensures that prompt length and inference compute remain strictly constant, regardless of the support bank size, while still conditioning the model on query-specific evidence. Consequently, the only trainable components are the GATv2 encoder, the dynamic context generator, and the layer-wise low-rank memory, leaving the foundational VLM and upstream embedding models entirely frozen.

4.2 GRAPH FORMULATION

In this subsection, we detail out the graph construction which is used at the next stages for building our dynamic contextual steering framework.

Part 1: Node construction

For any post $x_q = (t_q, I_q)$, where t_q is text and I_q is an optional image, we employ a frozen multilingual and modality aware encoder ϕ (QWEN 3-VL-EMBEDDING-2B in our case; refer to (Rizwan et al., 2026)) that maps the inputs to a shared retrieval space: $z_q = \phi_{\text{multilingual+multimodal}}(x_q) \in \mathbb{R}^{d_{\text{emb}}}$.

As an example, consider a textual post: "go back to your country you f**thy immigrant". This post is encoded by our chosen embedding model, producing $z_q \in \mathbb{R}^{d_{\text{emb}}}$. In a similar manner, all datasets training bank (whether textual or memes) are pre-encoded identically, producing a bank $\{z_1, \ldots, z_{|\mathcal{B}|}\} \subset \mathbb{R}^{d_{\text{emb}}}$ for that respective suite.

After generating the embeddings, we follow the procedure discussed in BERTOPIC[2] for creating nodes using the embeddings generated from the training bank. We build a graph with variable number of nodes depending on the labels of the dataset and that is why retrieving the top nearest neighbors based on cosine-similarity score upon training bank obtained embeddings would yield near-duplicate examples that provide redundant evidence. Therefore, HOLOGRAM uses a three-stage pipeline to balance relevance and diversity and it is closely based on BERTOPIC. First, we make clusters for our training bank by employing UMAP followed by K-MEANS++ clustering (done once per training bank): $\tilde{z}_i = \text{UMAP}(z_i) \in \mathbb{R}^{d'}, \quad \{P_1, \ldots, P_K\} = \text{K-MEANS++}(\{\tilde{z}_i\})$.

After this step, we obtain the representative nodes. The query's reduced embedding $\tilde{z}_q$ (of dimension d') is compared to the K cluster centroids. The M closest clusters are selected, and *within each selected cluster independently*, the S training examples most similar to z_q are identified. Importantly, this within-cluster ranking uses cosine similarity in the *original* d_{emb}-dimensional embedding space (not the UMAP space); UMAP is only used for cluster assignment. We mathematically present the selected pool for x_q as follows: $\text{Pool}_q = \bigcup_{k \in \text{Top-M}(\cos(\tilde{z}_q, \mu_k))} \left\{ \text{Top-S}_{i \in P_k} \cos(z_q, z_i) \right\}$. The operator $\text{Top-S}_{i \in P_k}$ returns the S *indices* within cluster P_k that maximize cosine similarity to the query. The union collects these indices across Top-M clusters into a single pool. This produces atmost $M \times S$ candidates and any duplicates are removed while preserving the order of first appearance.

While the above process ensures diversity to an extent, the following step further ensures better diversity. From the pool, $C{=}M \times S$ final candidates, the first candidate is simply the pool member most cosine-similar to the query. Each subsequent candidate is chosen by maximizing the MMR score (Goldstein & Carbonell, 1998) over all *remaining* (not-yet-selected) pool members d_i: $\text{MMR}(d_i) = \lambda \cdot \cos(z_q, z_i) - (1 - \lambda) \cdot \max_{d_j \in \mathcal{S}} \cos(z_i, z_j), d_j \neq d_i$ With an appropriate λ, relevance is properly weighted more than diversity, so the selected set stays query-relevant while avoiding near-duplicates. We therefore use manually tuned value of $\lambda = 0.8$ to maintain a perfectly oriented balance between similarity and diversity.

Finally, ***only during training***, we also ensure label-guaranteed augmentation. This step exists specifically to support the supervised contrastive loss (refer to Section 4.5). The loss requires both *same-label* and *different-label* candidates in the graph so it can pull same-class representations together and push different-class representations apart. However, MMR is label-blind; it selects for relevance and diversity without regard to class membership. For example, in practice a *toxic*

[2] https://maartengr.github.io/BERTopic/index.html

query's MMR candidates may all be *toxic*, leaving the SCL loss (Khosla et al., 2020) with no negative examples and therefore no gradient signal. To prevent this, after MMR selection, for each label y in the dataset's label set $\mathcal{Y}$, the k_{label} most query-similar examples of class y are considered for appending. Crucially, any example that MMR already selected is skipped, so only new examples are appended. This de-duplication is why C_{total} is a dynamic range rather than a fixed number: $C_{\text{total}} \in [C,\ C + |\mathcal{Y}| \times k_{\text{label}}]$. The lower bound C is reached when MMR happens to already include the k_{label} most query-similar examples for every label class; the label-guaranteed step finds them already in the pool and adds nothing. The upper bound $C + |\mathcal{Y}| \times k_{\text{label}}$ is reached when MMR selected candidates have zero overlap with the label-guaranteed candidates; in practice there is usually some overlap, so the actual count falls between the bounds. As such, the final number of nodes in our constructed graph are variable and is dependent on the number of dataset labels during training.

Part 2: Edges construction

From the above node construction process, we receive a total of C_{total} candidates and the query node using which we form a graph $\mathcal{G}_q$ with $N = C_{\text{total}} + 1$ total nodes. Node 0 is always the query; Nodes $1, \ldots, C_{\text{total}}$ are the candidates obtained (refer **Part-1**). Following types of edges are made:

1. Star edges (query $\leftrightarrow$ all candidates): For each candidate j, two directed edges are added: $0 \to j$ and $j \to 0$. This gives $2 \cdot C_{\text{total}}$ edges. Every candidate directly attends to the query and vice versa.

2. Candidate edges (candidate $\to$ candidate): For each candidate i, we connect exactly the k_{nn} most cosine-similar other candidates (excluding the query node and i itself) with directed edges $i \to j$. This gives $C_{\text{total}} \times k_{\text{nn}}$ number of edges. Note these edges are *uni-directional*: $i \to j$ does not imply $j \to i$. If both i and j are among each other's k_{nn} nearest candidates, both directed edges will exist; if not, only the one direction is present. These edges expose within-neighborhood structure which candidates are near-duplicates, which cover different aspects, and which represent competing label assignments.

3. Self-loops: Every node connects to itself with weight 1.0. This gives N edges and ensures each node retains its own representation during message passing.

Note that all edge weights are cosine similarity between the connected nodes' *frozen* embeddings (the same d_{emb}-dimensional vectors from **Part-1**): $w_{ij} = \cos(z_i, z_j) = \frac{z_i^\top z_j}{\|z_i\| \cdot \|z_j\|}$. In the above cases, we follow prior works that used BERTopic (Rizwan et al., 2025a; Bhaskar et al., 2026) and manually tune $d' = 15, K = 30, M = 4, S = 5, C = 10, k_{\text{label}} = 3$, and $k_{\text{nn}} = 6$.

4.3 Generating Few-shot Representative Support Embeddings

The constructed graph in previous sub-section is then processed by a single GATv2 layer that refines all node representations simultaneously. Unlike standard GAT, GATv2 computes a dynamic attention ranking that depends on both source and target node features and thus is the reason we employ it for our use case. We use the default code from Torch Geometric with the number of heads $H{=}4$. The equations used for representing the network are exactly the same as presented in GATv2 paper (Brody et al., 2022) and in the official code implementation[3]; therefore we skip those here. After passing our graphs through this network, we get the output of the shape of our initial embedding dimension itself, i.e., $\mathbb{R}^{d_{\text{emb}}}$. On top of these obtained embeddings, we further add a linear projection to collapse the embedding dimension to $d_{\text{out}} < d_{\text{emb}}$, followed by LayerNorm for stable steering: $h_i = \text{LayerNorm}\big(W_{\text{proj}}\tilde{h}_i + b_{\text{proj}}\big) \in \mathbb{R}^{d_{\text{out}}}, \quad W_{\text{proj}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{emb}}}$. Here, $\tilde{h}_i$ are the embeddings of node i obtained from GATv2. In this case, we used $d_{\text{out}} = d_{\text{emb}}/4$. After GATv2 has been applied, all node representations live in $\mathbb{R}^{d_{\text{out}}}$. The query representation h_0 and candidate representations $h_1, \ldots, h_{C_{\text{total}}}$ are L2-normalized, and candidates are ranked by cosine similarity to the query: $\text{score}(i) = \cos(h_0, h_i) = \frac{h_0^\top h_i}{\|h_0\| \cdot \|h_i\|}, \quad \mathcal{S}_q^{(k)} = \text{Top-k}_{i \in \{1,\ldots,C_{\text{total}}\}} \text{score}(i)$. The top-$k$ candidates (with $k \in \mathcal{K} = \{2, 4, 8\}$ used for joint few-shot training (refer to Section 4.5) and a single chosen k at inference (refer to Section 5)) become the selected support set. Their GATv2 representations $\{h_j \in \mathbb{R}^{d_{\text{out}}} : j \in \mathcal{S}_q^{(k)}\}$ are used to generate the low-rank context vectors (refer to Section 4.4). Although these examples are selected, they are *not* placed in the prompt; the prompt still remains zero-shot (refer to Section 5). The examples influence the VLMs exclusively through the context vector injection only.

[3] https://pytorch-geometric.readthedocs.io/en/2.6.1/generated/torch_geometric.nn.conv.GATv2Conv.html

4.4 LOW-RANK CONTEXT VECTOR AND INFERENCE TIME STEERING OF THE VLMS

MOTIVATION: We now discuss the motivation behind our proposal of low-rank context vector injection. In the PREFIX MEMORY-TUNING (PMT) paper (Wang et al., 2026), authors show the decay of PREFIX-TUNING (Li & Liang, 2021) as the prompt size varies. As a resolution, they propose a global memory matrix M which learns steering on the training suite. At inference, this one global matrix M is passed at each layer to steer the activation space. We argue that to handle complex heterogeneous toxic content moderation, one global matrix that steers the activation for all layers of the VLM decoder would fail. Therefore, we propose a low-rank decomposition matrix at each layer of the decoder for robust steering. Further, their approach (or even any other prior approaches like LIVE (Peng et al., 2024)) do not have the capability to include query-conditioned context at inference time which further limits their dynamic inclusion capability. This is where we use GATV2 retrieved top-k support examples to pass it as context to our low-rank matrices. Furthermore, HOLOGRAM employs a unified prompt template across all moderation tasks, varying only the natural-language task descriptions and label definitions (see Appendix B) for exact prompts and Section 5 for dataset statistics). By dynamically steering a frozen VLM rather than updating its parameters, HOLOGRAM seamlessly unifies multimodal reasoning, multilinguality, and diverse label taxonomies within a single, computationally efficient activation steering framework.

IMPLEMENTATION: Each of the k support representations obtained from GATV2 in the previous sub-section are first projected to VLM's hidden size. This step crucially handles heterogeneity and maps different dataset suites to a homogeneous contextual space: $x_j = W_{\text{inp}} h_j + b_{\text{inp}} \in \mathbb{R}^{d_{\text{hidden}}}, \quad W_{\text{inp}} \in \mathbb{R}^{d_{\text{hidden}} \times d_{\text{out}}}$. Finally, we apply a two-layer MLP with a TANH gating to produce *each-layer low-rank context vectors*: $c_j = W_2 \tanh(W_1 x_j + b_1) + b_2 \in \mathbb{R}^{L \cdot r}$, where $W_1 \in \mathbb{R}^{d_{\text{mid}} \times d_{\text{hidden}}}$, $W_2 \in \mathbb{R}^{(L \cdot r) \times d_{\text{mid}}}$, and $d_{\text{mid}} = 512$. Therefore, for each jth shot example we get context vectors of dimension $\mathbb{R}^r$ for each layer $l \in L$ in the decoder of VLMs. Finally, to get a combined context vector over all k-shots, all k supports are summed element-wise, which is the simplest possible aggregation: $\bar{c}_\ell = \sum_{j \in \mathcal{S}_q^{(k)}} c_{j,\ell} \in \mathbb{R}^r, \quad \ell = 1, \ldots, L.$

Layer-wise low-rank matrix computation: This is where we depart from PMT paper's single global matrix M and propose a layer-aware memory matrix M_l. Since, this could have come with the additional parameter bottleneck, we therefore decompose it as two low-rank matrices and insert context vector in between them for simultaneous dual benefit: contextual enhancement and parameter efficiency. For our experiments, we use rank $r = 64$ and for each decoder layer ℓ, the two learnable low-rank factors have dimensions $A_\ell \in \mathbb{R}^{d_q \times r}$ and $B_\ell \in \mathbb{R}^{r \times d_{\text{hidden}}}$ and are combined with the graph-embedding generated context vector: $M_\ell = A_\ell \cdot diag(\bar{c}_\ell) \cdot B_\ell = \underbrace{(\bar{c}_\ell \cdot A_\ell)}_{\text{scaled left factor}} \cdot B_\ell \in \mathbb{R}^{d_q \times d_{\text{hidden}}}$, where the first and second parts of the equations are equal. The second part is just an efficient way to compute M_l compared to the first part. Further, in the first part of the equation, $diag(\bar{c}_\ell)$ represents the diagonalization of context vector and the second part optimizes this computation by using broadcasting as scaled left factor multiplication. Here, $d_q = n_{\text{heads}} \times d_{\text{head}}$.

Similar to PMT paper, The L memory matrices $\{M_1, \ldots, M_L\}$ are injected into the frozen VLM using PYTORCH forward hooks and no VLM parameters are modified. For each decoder layer ℓ, at every token position t in the sequence, the modified attention output is: $o'_{\ell,t} = o_{\ell,t} + \underbrace{\text{ELU}(q_{\ell,t})}_{\in \mathbb{R}^{d_q}} \cdot \underbrace{M_\ell}_{\in \mathbb{R}^{d_q \times d_{\text{hidden}}}} \in \mathbb{R}^{d_{\text{hidden}}}$. Note that this is the same equation as PMT paper with single global matrix M replaced with layer-wise matrices M_l. The residual is therefore added after the attention output projection (o_{proj} output) but before the residual connection and feed-forward network. The frozen attention mechanism computes normally; the context vector dynamically adds a correction that shifts the representation toward the label the graph evidence supports.

4.5 TRAINING OBJECTIVE AND LOSS FUNCTIONS

Training optimizes the GATV2 encoder, the context vector generator, and the low-rank factors $\{A_\ell, B_\ell\}_{\ell=1}^{L}$ jointly. The VLM and embedding model remain frozen. The loss has two terms as discussed below.

(I) Supervised contrastive loss (SCL): As we discussed earlier in Section 4.2 and 4.3 this loss is applied over the GATV2 representations of *all* candidates (not just the top-k) in the current query's graph. Since we want the graph representation to be very robust, we hard mine the most

Table 1: Dataset statistics: our setup leads to a total of nine configurations. In column ***modality***, T: textual posts, M: memes, and in column ***# train*** and ***# test***, h: hateful, o: offensive, n: normal, t:toxic, p: profane, and d: dangerous.

dataset	language	modality	# labels	# train	# test
HATEXPLAIN	ENGLISH	T	3 2	h: 4,748, o: 4,384, n: 6,251 t: 9,132, n: 6,251	h: 594, o: 548, n: 782 t: 1,142, n: 782
AMHARIC	AMHARIC	T	3 2	h: 5,283, o: 2,240, n: 4,477 t: 7,523, n: 4,477	h: 702, o: 264, n: 535 t: 966, n: 535
HASOC	HINDI	T	4 2	h: 556, o: 676, p: 1,237, n: 2,196 t: 2,469, n: 2,196	h: 190, o: 197, p: 218, n: 713 t: 605, n: 713
BHM	CODE-MIXED BENGALI	M	2	h: 2,117, n: 3,641	h: 266, n: 445
TOXICTAGS	ENGLISH	M	4 2	h: 1,470, d: 1,847, o: 686, n: 1,297 t: 4,003, n: 1,297	h: 282, d: 472, o: 97, n: 149 t: 851, n: 149

informative pairs: **i) hard positives**: Same-label candidates whose post-GATv2 cosine similarity is below $\tau_{\text{pos}} = 0.75$. These are same-class examples that are hard to pull together, **ii) hard negatives**: Different-label candidates whose post-GATv2 cosine similarity is above $\tau_{\text{neg}} = 0.5$. These are different-class examples that are hard to push apart, and **iii) fallback**: If no candidates pass a threshold, the top-k_{fb}=3 least similar positives or most similar negatives are used.

Therefore, for each node i: $\mathcal{L}_{\text{SCL}}^{(i)} = -\frac{1}{|\mathcal{P}_i|}\sum_{p\in\mathcal{P}_i}\log\frac{\exp(h_i^\top h_p/Temp)}{\sum_{j\in\mathcal{P}_i\cup\mathcal{N}_i}\exp(h_i^\top h_j/Temp)}$, (Khosla et al., 2020). The total SCL averages over all anchors that have both positives and negatives. Here in the above equation, $\mathcal{P}_i$ represent the most informative pairs for node i, and $Temp = 0.1$.

(II) Cross-entropy loss (CE): To train a single model capable of handling varying amounts of support evidence during inference (refer to Section 5), we optimize over a predefined set of shot counts $\mathcal{K}$ (in our experiments, $\mathcal{K} = \{2, 4, 8\}$) simultaneously. For each training query and each $k \in \mathcal{K}$, context is registered using the top-k GATv2 representations, and the frozen VLM executes a forward pass with the ground-truth label appended to the prompt. The cross-entropy is computed *exclusively* over the label tokens (prompt tokens are masked). The final objective is aggregated using a shot-proportional weighted average: $\mathcal{L}_{\text{CE}} = w_y \cdot \frac{\sum_{k\in\mathcal{K}} k\cdot\mathcal{L}_{\text{CE}}^{(k)}}{\sum_{k\in\mathcal{K}} k}$, where $w_y = \frac{|\mathcal{B}|}{|\mathcal{Y}|\cdot n_y}$ is the class weight for the true label y (Swain et al., 2026). The term $\mathcal{L}_{\text{CE}}^{(k)}$ denotes the loss computed when the context memory M_ℓ is generated using exactly the top-k retrieved GATv2 representations.

(III) Combined objective: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{SCL}}$. The CE term teaches the context to steer the frozen decoder toward the correct label; the SCL term teaches the graph encoder to produce representations where same-class examples cluster and different-class examples separate.

5 EMPLOYED DATASETS, MODELS AND EXPERIMENTAL SETUP

(I) EMPLOYED DATASETS – We employ three textual and two meme hate speech datasets. To ground heterogeneity, these datasets ensure coverage across diverse languages, modalities, and label granularities (binary and fine-grained), yielding ***nine distinct configurations***. The text datasets are ENGLISH (HATEXPLAIN (Mathew et al., 2021)), AMHARIC (Ayele et al., 2023), and HINDI (HASOC (Mandl et al., 2019)); the meme datasets are CODE-MIXED BENGALI (BHM (Hossain et al., 2024)) and ENGLISH (TOXICTAGS (Swain et al., 2026)). The choice of these languages reflects the authors' regional affiliations. Detailed dataset configurations and splits are in Table 1.

(II) EMPLOYED MODELS – (a) VLMs and embeddings: We use the 2 and 8 billion instruction-tuned QWEN 3-VL (Bai et al., 2025) and INTERN 3.5-VL (Wang et al., 2025) models. QWEN 3-VL-EMBEDDING-2B (Zhang et al., 2025) supplies multilingual-multimodal embedding representations. Prior to using all these models, we agree to their usage terms on HUGGINGFACE. **(b) baselines**: We evaluate HOLOGRAM (at 2, 4, and 8 shots) against zero-shot, chain-of-thought, and similarity-retrieved ICL (QWEN 3-VL-EMBEDDING-2B) at corresponding shot lengths, alongside two vector-based steering baselines: LIVE (Peng et al., 2024) and PMT (Wang et al., 2026). Few-shot support sets are drawn from training data, with cross-split duplicates removed via perceptual hashing. For metric, per prior works (Rizwan et al., 2026; Bhaskar et al., 2026), we use Macro-F1 scores.

(III) EXPERIMENTAL SETUP – (a) hyperparameters: We optimize all trainable parameters (GATv2, context generator, and low-rank factors) using ADAMW (learning rate: 2×10^{-5}) with gradient accumulation over U=32 queries. Further, zero-initializing the B_ℓ matrix ensures $M_\ell = \mathbf{0}$ initially, precisely preserving the frozen VLM's original behavior. The context vector $\bar{c}_\ell$ is strictly *input-*

Table 2: **Macro-F1 scores across the experimental suite:** Here, **zs**: zero-shot, **CoT**: chain of thoughts, **PMT**: prefix memory tuning. **ICL**, **HOLOGRAM** are evaluated for 2, 4, and 8 shots. Beside each dataset, the tuple denotes the modality (`T` for textual dataset and `M` for meme datasets) and the number of classification labels. The two best methods are highlighted in each case, i.e., for each (model, dataset) pair. The best method in each case is marked **green** and the second best one is **highlighted**. For HOLOGRAM, we also put relative increase/decrease in performance compared to the second/first best method below the corresponding Macro-F1 score. Here, **Q 3-VL**: QWEN 3-VL, and **I 3.5-VL**: INTERN 3.5-VL.

models	datasets	zs	ICL			CoT	PMT	LIVE	HOLOGRAM (ours)		
			2	4	8				2	4	8
Q 3-VL (2B)	HATEXPLAIN *(T, 3)*	34.69	44.84	47.93	50.24	41.11	**62.75**	50.23	66.27	**67.88** *(+5.13)*	67.82
	AMHARIC *(T, 3)*	33.77	40.66	40.21	38.07	34.08	**57.35**	51.08	54.5	**59.72** *(+2.37)*	58.89
	HASOC *(T, 4)*	36.18	37.38	43.03	**45.36**	31	40.39	42.73	54.25	**59.43** *(+14.07)*	57.34
	HATEXPLAIN *(T, 2)*	50.42	61.98	59.71	61.51	64.22	**76.77**	67.61	78.25	**79.06** *(+2.29)*	77.72
	AMHARIC *(T, 2)*	43.5	53.56	49.14	46.41	56.01	57.96	**60.31**	62.8	66.79	**68.19** *(+7.88)*
	HASOC *(T, 2)*	56.28	66.28	**68.12**	67.76	67.98	64.44	62.48	79.83	82.74	**82.98** *(+14.86)*
	BHM *(M, 2)*	52.8	**63.6**	61.78	61.27	40.6	43.84	60.8	66.41	**68.5** *(+4.9)*	67.04
	TOXICTAGS *(M, 4)*	26.95	36.44	35.86	34.01	16.72	**52.45**	30.97	52.23	**56.88** *(+4.43)*	56.16
	TOXICTAGS *(M, 2)*	53.89	**60.87**	57.67	54.68	23.85	60.32	60.18	57.36	62.93	**65.16** *(+4.29)*
I 3.5-VL (2B)	HATEXPLAIN *(T, 3)*	44.9	45.95	48.87	50.49	41.34	**65.17**	45.46	67.46	**69.29** *(+4.12)*	65.81
	AMHARIC *(T, 3)*	34.19	39.46	38.68	40.35	26.59	**56.12**	52.94	54.85	59.45	**61.24** *(+5.12)*
	HASOC *(T, 4)*	27.69	35.38	45.59	**46.91**	30.66	29.7	38.7	56.74	**61** *(+14.09)*	58.4
	HATEXPLAIN *(T, 2)*	55.78	67.42	68.48	69.52	64.46	**76.56**	70.74	78.08	80.21	**80.25** *(+3.69)*
	AMHARIC *(T, 2)*	24.54	60.03	59.22	60.78	41.22	**66.2**	54.64	66.12	70.96	**71.42** *(+5.22)*
	HASOC *(T, 2)*	39.09	71.85	73.89	**75.34**	68.19	74.19	72.96	78.52	81.24	**81.46** *(+6.12)*
	BHM *(M, 2)*	42.12	**62.39**	61.69	62.25	49.28	61.09	44.84	61.8	**66.57** *(+4.18)*	61.46
	TOXICTAGS *(M, 4)*	21.91	30.42	37.64	38.27	18.09	**50.11**	37.58	43.85	52.13	**55.12** *(+5.01)*
	TOXICTAGS *(M, 2)*	20.41	52.49	51.12	56.82	34.86	**63.55**	53.04	54.2	59.25	**65.88** *(+2.33)*
Q 3-VL (8B)	HATEXPLAIN *(T, 3)*	50.18	51.74	57.21	60.28	51.68	**62.69**	58.87	68.34	**68.53** *(+5.84)*	64.67
	AMHARIC *(T, 3)*	34.53	51.14	52.15	52.32	39.84	53.78	**55.29**	59.88	**60.65** *(+5.36)*	58.5
	HASOC *(T, 4)*	45.3	51.49	54.44	**54.8**	32.07	34.37	54.19	63.11	**64.13** *(+9.33)*	61.65
	HATEXPLAIN *(T, 2)*	60.12	63.05	68.57	72.35	68.34	**74.42**	72.49	80.68	**81.29** *(+6.87)*	81.03
	AMHARIC *(T, 2)*	50.74	64.58	64.72	**66.88**	44.77	52.34	65.9	70.77	**74.19** *(+7.31)*	73.9
	HASOC *(T, 2)*	78.61	76.81	77.89	77.75	**82.26**	31.46	75.26	**84.56** *(+2.3)*	84.51	83.82
	BHM *(M, 2)*	44.49	58.3	61.83	64.82	53.11	**66.75**	63.14	70.12	**71.77** *(+5.02)*	71.48
	TOXICTAGS *(M, 4)*	28.57	42	47.38	**49.33**	23.52	33.95	46.25	53.86	**58.07** *(+8.74)*	54.9
	TOXICTAGS *(M, 2)*	46.04	59.32	59.57	**61.29**	35.43	21.55	59	61.99	66.65	**66.78** *(+5.49)*
I 3.5-VL (8B)	HATEXPLAIN *(T, 3)*	50.75	54.17	58.5	61.35	47.79	**62.39**	61.49	68.46	**68.61** *(+6.22)*	66.62
	AMHARIC *(T, 3)*	31.74	39.85	46.79	**51.19**	32.09	51.04	49.93	61.06	**65.81** *(+14.62)*	65.26
	HASOC *(T, 4)*	37.17	45.41	49.02	50.35	30.76	29.1	**54.85**	61.96	62.59	**62.74** *(+7.89)*
	HATEXPLAIN *(T, 2)*	64.25	67.97	72.77	**74.87**	63.02	73.95	74.25	**80.84** *(+5.97)*	80.25	79.86
	AMHARIC *(T, 2)*	31.1	62.43	63.76	**64.61**	37.45	53.63	63.98	71.97	73.43	**74.33** *(+9.72)*
	HASOC *(T, 2)*	76.78	75.65	77.83	**78.14**	77.25	31.46	76.11	83	**83.84** *(+5.7)*	83.62
	BHM *(M, 2)*	40.45	49.64	50.53	51.08	30.38	**54.28**	52.45	68.8	70.28	**72.7** *(+18.42)*
	TOXICTAGS *(M, 4)*	26.64	30.31	34.95	**42.26**	18.73	40.23	41.63	53.46	**57.11** *(+14.85)*	56.81
	TOXICTAGS *(M, 2)*	50.39	58.37	58.33	59.14	44.25	38.15	**59.31**	59.73	62.18	**65.31** *(+6)*

dependent, dynamically derived from the GATv2 representations of each query's specific support set. We train HOLOGRAM for three epochs, while the LIVE and PMT frameworks are trained for five. As detailed in Section 6, HOLOGRAM outperforms all baselines despite this reduced training epochs. **(b) stratified k-fold training**: The training bank $\mathcal{B}$ is split into n_{folds}=4 stratified folds, shuffled each epoch. For each fold, 75% serve as the retrieval bank exclusively for UMAP, clustering, and candidate selection. and the rest 25% supplies training queries that undergo the full pipeline (candidate selection → graph → GATv2 → context vectors → VLM forward → loss). This structure strictly prevents the model from retrieving examples from its active query partition. **(c) inference**: At test time, the pipeline is identical to training with two differences: (i) the full training bank $\mathcal{B}$ is used for retrieval, and (ii) no addition of nodes for label balancing (refer to Section 4.2); prompt remains zero-shot and support examples reach the decoder only through the context vector injection.

6 RESULTS

We present our comprehensive experimental results in Table 2. HOLOGRAM substantially and consistently outperforms all baseline methods across all datasets and label granularities. In the following subsections, we provide a detailed analysis of these results.

Comparison to non-training baselines: We evaluate three non-training baselines: (i) zero-shot, (ii) chain-of-thought, and (iii) in-context learning (ICL) at 2, 4, and 8 shots (Rizwan et al., 2026). For ICL, we append similarity-ranked examples retrieved via QWEN 3-VL-EMBEDDING-2B. HOLOGRAM substantially outperforms these baselines across all datasets. While ICL remains the strongest baseline (next to HOLOGRAM) (ranking second in 16 of 36 settings, Table 2), HOLOGRAM achieves absolute gains of up to 14% on 2B parameter models in some configurations. These results demonstrate that standard prompting struggles with heterogeneity, a limitation HOLOGRAM successfully resolves.

Comparison to training baselines: We compare against two recent activation-space steering methods: PMT and LIVE. Unlike ICL and CoT, these avoid extensive prompt-based context. For a fair comparison, we omit graph-ranked few-shot examples from HOLOGRAM's prompt (i.e. we do not append few-shot in prompts), passing them exclusively as context vectors only. HOLOGRAM consistently outperforms these contemporary techniques. Contrasting directly with PMT, our findings demonstrate that per-layer low-rank contextual memory adapters steer representations substantially better than a single memory block applied across all frozen VLM layers.

Handling heterogeneity: Our primary motivation was designing a unified steering framework to manage the inherent heterogeneity of toxic content moderation. As shown in Table 2, HOLOGRAM achieves state-of-the-art performance, securing the top result across all 36 (model, dataset) configurations. It robustly adapts to multilinguality, multimodality, and diverse label granularities, notably yielding substantial gains (up to 14%) on non-ENGLISH languages like AMHARIC and HINDI. Furthermore, its pronounced effectiveness in multi-granular classification (i.e., datasets exceeding two labels) establishes it as a highly versatile, homogenizing framework for diverse moderation scenarios.

VLMs behavior: We observe an interesting trend regarding the optimal number of contextual examples: 4-shot-HOLOGRAM outperforms both the 2-shot and 8-shot configurations in 20 out of 36 (model, dataset) pair (with 8-shot being optimal in 14 cases, and 2-shot in only two). While the exact mechanism driving this behavior warrants future investigation, it highlights a potential saturation point for context vector utility. Surprisingly, we also observe that the 2B parameter variants approach their 8B counterparts when equipped with HOLOGRAM. This demonstrates that our framework not only mitigates the heterogeneity problem but also paves the way for highly effective content moderation using smaller and less compute-intensive VLMs.

Due to paucity of space, we put further experiments and extensive analysis in Appendices C and D.

7 CONCLUSION

We introduced HOLOGRAM, a graph-conditioned context vector guidance framework for a problem that has previously been fragmented by language, modality, and label taxonomy. Its defining idea is simple: the local examples used to understand a social-media post should form a relational representation, and that representation should control both the evidence shown to a model and the way its frozen computation is adapted. The recorded experiments show substantial improvements for QWEN 3-VL and INTERN 3.5-VL models when compared to strong existing frameworks. Also, HOLOGRAM offers a path from heterogeneous classifiers toward unified moderation systems.

AI USE STATEMENT

In this work, we used generative AI tools solely to assist in editing and refining the language of this research paper. We have not used generative AI tools to produce synthetic data, formulate theoretical models, write code, design experiments, or write mathematical proofs. We have carefully reviewed and edited all AI-assisted writing to ensure factual correctness, clarity, and strict adherence to our original ideas. As authors, we take full responsibility for the entirety of the contents within this submission.

ETHICS STATEMENT

Semantic retrieval can amplify annotation bias or spuriously associate protected-group mentions with harmful labels. A deployed system should use calibrated abstention, native-language review, appeal mechanisms, and continuing false-positive audits by target group. Retrieval banks may contain personally identifying, copyrighted, or disturbing content; deployment therefore requires minimal retention, access control, and careful handling of demonstrations. Released artifacts will follow dataset licenses and avoid redistributing restricted images or unnecessary hateful examples. HOLOGRAM is intended to assist trained moderators, not to autonomously punish users or infer protected attributes.

REPRODUCIBILITY STATEMENT

We have provided significant details while discussing our architecture and while presenting the experimental setup. We will make our paper's code publicly available upon acceptance of the work.

REFERENCES

Sayantan Adak, Somnath Banerjee, Rajarshi Mandal, Avik Halder, Sayan Layek, Rima Hazra, and Animesh Mukherjee. Memesense: an adaptive in-context framework for social commonsense driven meme moderation. *arXiv preprint arXiv:2502.11246*, 2025.

Abinew Ali Ayele, Seid Muhie Yimam, Tadesse Destaw Belay, Tesfa Asfaw, and Chris Biemann. Exploring Amharic hate speech data collection and classification approaches. In Ruslan Mitkov and Galia Angelova (eds.), *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pp. 49–59, Varna, Bulgaria, September 2023. INCOMA Ltd., Shoumen, Bulgaria. URL https://aclanthology.org/2023.ranlp-1.6/.

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, 2025. URL https://arxiv.org/abs/2511.21631.

Paramananda Bhaskar, Naquee Rizwan, Daksh Jogchand, Saurabh Kumar Pandey, and Animesh Mukherjee. Fbhm: Functional benchmarking and steering of vlms for hateful meme detection, 2026. URL https://arxiv.org/abs/2605.31349.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

Minh Duc Bui, Katharina von der Wense, and Anne Lauscher. Multi^{3}Hate: Multimodal, multilingual, and multicultural hate speech detection with vision–language models. In Luis Chiruzzo, Alan Ritter,

and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9714–9731, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.490. URL https://aclanthology.org/2025.naacl-long.490/.

Rui Cao, Ming Shan Hee, Adriel Kuek, Wen-Haw Chong, Roy Ka-Wei Lee, and Jing Jiang. Pro-cap: Leveraging a frozen vision-language model for hateful meme detection. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, pp. 5244–5252, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701085. doi: 10.1145/3581783.3612498. URL https://doi.org/10.1145/3581783.3612498.

Jade Goldstein and Jaime Carbonell. Summarization: (1) using MMR for diversity- based reranking and (2) evaluating summaries. In *TIPSTER TEXT PROGRAM PHASE III: Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, pp. 181–195, Baltimore, Maryland, USA, October 1998. Association for Computational Linguistics. doi: 10.3115/1119089.1119120. URL https://aclanthology.org/X98-1025/.

Ming Shan Hee, Aditi Kumaresan, and Roy Ka-Wei Lee. Bridging modalities: Enhancing cross-modality hate speech detection with few-shot in-context learning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7785–7799, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.445. URL https://aclanthology.org/2024.emnlp-main.445/.

Ming Shan Hee, Shivam Sharma, Rui Cao, Palash Nandi, Preslav Nakov, Tanmoy Chakraborty, and Roy Ka-Wei Lee. Recent advances in online hate speech moderation: Multimodality and the role of large models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4407–4419, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.254. URL https://aclanthology.org/2024.findings-emnlp.254/.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9318–9333, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL https://aclanthology.org/2023.findings-emnlp.624/.

Eftekhar Hossain, Omar Sharif, Mohammed Moshiul Hoque, and Sarah M. Preum. Deciphering hate: Identifying hateful memes and their targets. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8347–8359, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.454. URL https://aclanthology.org/2024.acl-long.454/.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Prince Jha, Raghav Jain, Konika Mandal, Aman Chadha, Sriparna Saha, and Pushpak Bhattacharyya. MemeGuard: An LLM and VLM-based framework for advancing content moderation via meme intervention. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8084–8104, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.439. URL https://aclanthology.org/2024.acl-long.439/.

Prashant Kapil and Asif Ekbal. A survey on combating hate speech through detection and prevention in English. In Sobha Lalitha Devi and Karunesh Arora (eds.), *Proceedings of the 21st International Conference on Natural Language Processing (ICON)*, pp. 485–501, AU-KBC

Research Centre, Chennai, India, December 2024. NLP Association of India (NLPAI). URL https://aclanthology.org/2024.icon-1.57/.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The hateful memes challenge: detecting hate speech in multimodal memes. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *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, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Junyu Lu, Bo Xu, Xiaokun Zhang, Haohao Zhu, Kaichun Wang, Liang Yang, and Hongfei Lin. Is having rationales enough? rethinking knowledge enhancement for multimodal hateful meme detection. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, pp. 559–569, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715921. doi: 10.1145/3726302.3730014. URL https://doi.org/10.1145/3726302.3730014.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation*, pp. 14–17, 2019.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 14867–14875, 2021.

Yingzhe Peng, chenduo hao, Xinting Hu, Jiawei Peng, Xin Geng, and Xu Yang. LIVE: Learnable in-context vector for visual question answering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=QhRemVrZbG.

Naquee Rizwan, Paramananda Bhaskar, Mithun Das, Swadhin Satyaprakash Majhi, Punyajoy Saha, and Animesh Mukherjee. Exploring the limits of zero shot vision language models for hate meme detection: The vulnerabilities and their interpretations. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 19, pp. 1669–1689, 2025a.

Naquee Rizwan, Seid Muhie Yimam, Daryna Dementieva, Dr. Florian Skupin, Tim Fischer, Daniil Moskovskiy, Aarushi Ajay Borkar, Robert Geislinger, Punyajoy Saha, Sarthak Roy, Martin Semmann, Alexander Panchenko, Chris Biemann, and Animesh Mukherjee. HatePRISM: Policies, platforms, and research integration. advancing NLP for hate speech proactive mitigation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 16008–16022, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.824. URL https://aclanthology.org/2025.findings-acl.824/.

Naquee Rizwan, Subhankar Swain, Paramananda Bhaskar, Gagan Aryan, Shehryaar Shah Khan, and Animesh Mukherjee. See, explain, and intervene: A few-shot multimodal agent framework for hateful meme moderation, 2026. URL https://arxiv.org/abs/2601.04692.

Subhankar Swain, Naquee Rizwan, Vishwa Gangadhar S, Nayandeep Deb, and Animesh Mukherjee. Stemtox: From collaborative tags to fine-grained toxic meme detection via entropy-guided multi-task learning. *Transactions of the Association for Computational Linguistics*, 14:2281–2300, 08 2026. ISSN 2307-387X. doi: 10.1162/TACL.a.801. URL https://doi.org/10.1162/TACL.a.801.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

Haonan Wang, Brian K Chen, Siquan Li, Liang Xinhe, Hwee Kuan Lee, Kenji Kawaguchi, and Tianyang Hu. Prefixmemory-tuning: Modernizing prefix-tuning by decoupling the prefix from attention. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=LvUMpZE44r.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025. URL https://arxiv.org/abs/2508.18265.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. URL https://arxiv.org/abs/2506.05176.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2025. URL https://arxiv.org/abs/2310.01405.

A Limitations

We cover multiple languages and substantially broader than an English-only evaluation, but do not claim of universal cultural coverage. Language, culture, and moderation policy are related but not equivalent and the benchmarks also contain annotator disagreement and historically contingent definitions of hate which needs to be further explored (Rizwan et al., 2025b). Our architecture models the supplied policy; it does not determine which policy is socially legitimate and it lays in the hands of moderators. The graph-encoder represents local relations among content examples, not users or social ties, and must not be interpreted as a causal account of intent or diffusion.

B Employed Prompts

All methods share one dataset-agnostic prompt template. Only following fields change: the task name `{name}`, the label list `{labels}`, and the label definitions `{definitions}`. The output instruction `{instruction}` is identical for all datasets. For meme datasets, every user turn that carries an input consists of the image followed by the OCR extracted text; for textual datasets, it consists of the post alone. All prompts are rendered with the chat template of the respective VLM using HuggingFace APIs.

zero-shot, PMT, LIVE, and HOLOGRAM

System: You are a Content Moderation Expert and are tasked with {name} into {labels} labels
Consider the following definition of labels:
{definitions}
Strictly follow the following output instructions:
{instruction}

User: {query input}
Assistant:

HOLOGRAM uses this prompt with its retrieved support examples never entering the prompt and reach the VLM only through the context vectors (Section 4.4).

ICL with k examples

System: You are a Content Moderation Expert and are tasked with {name} into {labels} labels
Consider the following definition of labels:
{definitions}
Strictly follow the following output instructions:
{instruction}

User: {example 1 input}
Assistant: {example 1 label}
... *(repeated for k examples)*
User: {example k input}
Assistant: {example k label}

User: {query input}
Assistant:

The k in-context examples are the training examples most similar to the query under QWEN 3-VL-EMBEDDING-2B, given as preceding user–assistant turns.

chain-of-thoughts prompt *(two passes)*

Pass 1: generate the reasoning

System: You are a Content Moderation Expert and are tasked with {name} into {labels} labels
Consider the following definition of labels:
{definitions}

User: {query input}
Assistant: Let's think step by step.
Model output: {reasoning}

Pass 2: generate the label

Input: the complete pass-1 prompt, followed by
{reasoning}
Therefore, among {labels}, the final label is:
Model output: {label}

In pass 1, the assistant turn is opened with the reasoning trigger and the model writes its reasoning. In pass 2, this reasoning and the answer trigger are appended to the same assistant turn, and the model writes the final label. The prompt omits the output instruction, since the two trigger sentences control the reasoning and the answer format.

C CAN A SINGLE MODEL WORK FOR COMBINED SUITES?

The evaluations in Table 2 trains each pair of (model, dataset) for HOLOGRAM. However, in a real world scenario, deployed moderation system faces heterogeneity all at once. We therefore ask

Table 3: **Macro-F1 of a single model trained on a combined suite:** The overall (for each combined suite) best method is marked **green** and the second best is **highlighted**. Here, **Q 3-VL**: QWEN 3-VL, and **I 3.5-VL**: INTERN 3.5-VL.

models	datasets	HOLOGRAM (ours)		
		2	4	8
Q 3-VL (2B)	MULTILINGUAL *(T, 2)*	75.40	**76.30**	75.14
	MULTIMODAL *(M, 2)*	68.64	70.75	**72.06**
I 3.5-VL (2B)	MULTILINGUAL *(T, 2)*	74.50	76.56	**76.98**
	MULTIMODAL *(M, 2)*	67.37	71.44	**72.76**

whether a *single* HOLOGRAM model, trained on a mixture of languages or a mixture of meme sources remains effective when the heterogeneity lies within its own training and test data rather than across separate models. We answer these questions in this section.

Combined suites: We construct two different combined suites from the datasets in Table 1 and specifically consider only binary cases because the fine-grained label sets are not aligned (for example: HASOC has *profane* labels as well while HATEXPLAIN does not have; TOXICTAGS has *dangerous*, and *offensive* samples while BHM does not have). These categories follow different annotation policies and have no one-to-one correspondence, so merging them into a shared fine-grained taxonomy would impose an arbitrary mapping. The binary distinction between *toxic* and *normal* content is the only label space that all sources share; every non-normal class is therefore mapped to *toxic*, exactly as for the binary configurations of Table 2. Each example keeps its source and language as metadata.

MULTILINGUAL (T, 2) combines HATEXPLAIN (ENGLISH), AMHARIC, and HASOC (HINDI). **Training**: from each source's training split we draw 5,000 posts by stratified sampling on the binary label; which preserves each source's toxic/normal ratio and prevents the largest source from dominating. HASOC has fewer training posts (4,665) and is used in full, giving a total of 14,665 training samples (58.5% are *toxic*). **Testing**: the complete test splits are concatenated without sampling, giving 4,743 test posts (57.2% are *toxic*).

MULTIMODAL (M, 2) combines BHM (CODE-MIXED BENGALI memes) and TOXICTAGS (ENGLISH memes). **Training**: we jointly extract 5,000 memes from both datsets in a stratified manner; which yields 2,606 BHM and 2,394 TOXICTAGS memes (55.3% are *toxic*). **Testing**: the complete test splits are concatenated, giving 1,711 test memes (65.3% are *toxic*).

Since the combined test sets are the same test splits used for the single-dataset evaluations, per-source results on the combined suites remain directly comparable to them. All other settings follow Section 5 and we report the results for the 2B VLMs only (as they are on-par with their 8B counter-part).

RESULTS: As presented Table 3, HOLOGRAM achieves a competitive score of **76.98%** on MULTILINGUAL and **72.76%** on MULTIMODAL suites for INTERN 3.5-VL 2B (QWEN 3-VL 2B nearly follows). These results further signify that HOLOGRAM does not only work with single configuration training, but has the potential to jointly mitigate heterogeneous behavior of the real world social media toxic content moderation. Further, here 8-shots work better than 4-shots, signifying that complex tasks require more representations of graph-modeled few-shot representative embeddings.

D ANALYSIS

While Section 6 establishes the empirical superiority of HOLOGRAM over competing baselines, this section investigates the underlying mechanisms driving this success. The architectural design of HOLOGRAM is predicated on four core hypotheses: (1) the GATv2 encoder effectively contextualizes each support example relative to its retrieving query; (2) the resulting context vectors encode task-relevant structural priors, despite the absence of explicit labels during inference; (3) the VLM actively utilizes this semantic content rather than merely reacting to the addition of a generic residual vector; and (4) the injected context steers the frozen VLM's computation rather than bluntly overriding its output. We empirically validate each of these principles below. All analyses are conducted strictly at inference time, reusing the trained 2B checkpoints (QWEN 3-VL 2B and INTERN 3.5-VL 2B) without any further parameter updates. Evaluations are performed across the *full* test sets using the exact retrieval pipeline detailed in Section 5.

Key Findings

☞ **Query-conditioned support representations**: The same training example is represented differently based on its retrieving query. We find that 34–55% of the variance in GATv2 outputs is attributable to this retrieval context, which strongly correlates with query similarity (Spearman's $\rho = 0.39$–0.58).
☞ **Implicit label clustering in context vectors**: Although explicit labels never reach the context builder during inference, target classes cluster 5.7–14.2× more tightly within the context vectors than within the frozen query embeddings across HASOC, TOXICTAGS, and the combined multimodal suite.
☞ **Semantic content over raw capacity**: Replacing retrieved representations with random vectors of matching mean and scale yields marginal Macro-F1 gains ($+0.10$ to $+7.31$). In contrast, the true semantic context drives substantial improvements ($+14.27$ to $+40.69$) consistently across all languages.
☞ **Deep network steering**: Context injection shifts the final-layer attention output by 21–67%. However, only 3.2–4.4% of this shift lies in the subspace directly written by the final layer (compared to 3.1% for a random subspace), demonstrating that steering effects compound fundamentally from injections at earlier layers.

D.1 SETUP

We analyze four distinct (dataset, checkpoint) configurations per model. Two comprise the fine-grained suites from Table 2: HASOC (T, 4) and TOXICTAGS (M, 4). The remaining two are the combined binary suites, MULTILINGUAL (T, 2) and MULTIMODAL (M, 2), from Appendix C, each featuring a single model trained across its respective source datasets. These combined suites expose the heterogeneity discussed in Section 1 within a unified model, enabling us to examine cross-lingual context behavior. Retrieval, graph construction, and top-k selection strictly mirror the inference setup (Section 5); decoding remains greedy. In all plots, HASOC is labeled as HINDI.

D.2 DOES THE GATV2 ENCODER CONDITION SUPPORT REPRESENTATIONS ON THE QUERY?

If the GATv2 encoder acted merely as a fixed projection, a given training example would yield an identical representation h_j (Eq. 4.3) for every query that retrieves it, reducing the context vector to a simple lookup table over the training bank. In practice, retrieval naturally reuses examples: 2,241–8,080 training examples are retrieved by at least two test queries, averaging 3.9–5.5 retrievals each. For every repeatedly retrieved example, we compare its GATv2 outputs across different queries (*same example*; up to 20 pairs per example) against the outputs of two distinct examples (*different examples*; an equal number of randomly drawn pairs). Because the final LAYERNORM imparts a large shared component to all outputs (uncentered cosines ≈ 0.999), we subtract the mean output prior to computing cosine similarities, applying the same centering to query embeddings. We report the *context share* $(1 - \bar{s}_{\text{same}})/(1 - \bar{s}_{\text{diff}})$, representing the fraction of variance among GATv2 outputs originating from the retrieval context rather than the example's inherent identity, alongside the Spearman's correlation between query similarity and the similarity of the example's representations under those queries.

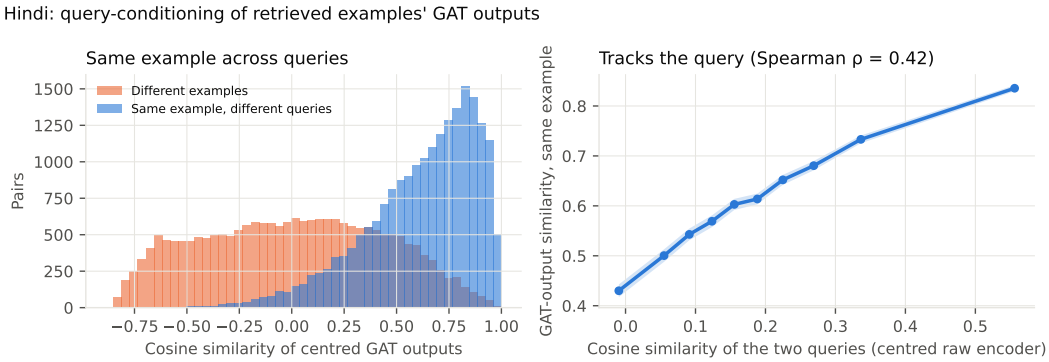
Table 4 and Figure 2 demonstrate that an example's representations across different queries remain significantly more similar than those of distinct examples (0.45–0.66 vs. ≤ 0.02), confirming that the example retains its core identity. However, 34–55% of the representational variance is dictated by the retrieval context, and these representations shift in tandem with query similarity ($\rho = 0.39$–0.58). Consequently, the input to the context builder is dynamically assembled per query rather than statically retrieved. It is important to note that the retrieval context encompasses both the query and its retrieved neighbors; the specific contributions of these two factors are not isolated in this analysis.

D.3 DO THE CONTEXT VECTORS CARRY LABEL STRUCTURE?

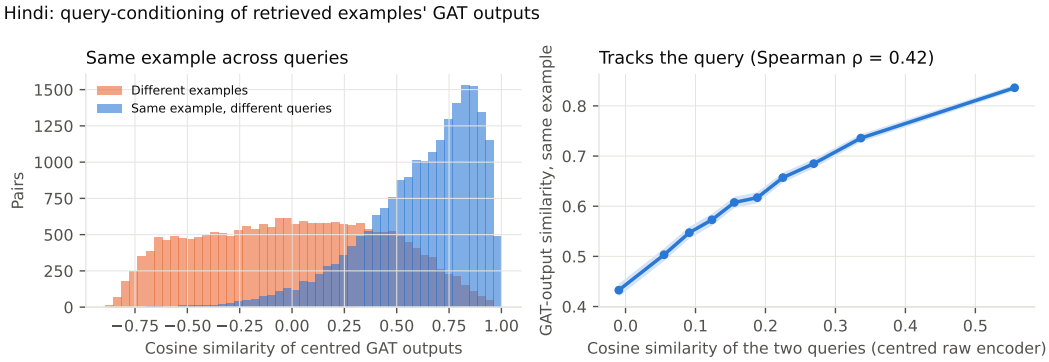
Given the trained low-rank factors $\{A_\ell, B_\ell\}$, any query-specific information in M_ℓ is entirely encapsulated by the aggregated context vectors $\bar{c}_\ell$ (Section 4.4). We therefore concatenate them across all layers, $\bar{c}_q = [\bar{c}_1; \ldots; \bar{c}_L] \in \mathbb{R}^{L \cdot r}$, and compare them against the frozen embedding z_q of the

Table 4: **Query conditioning of GATv2 support representations. Repeated**: training examples retrieved by ≥ 2 test queries; **Per ex.**: mean retrievals per repeated example; **Same / Diff**: mean cosine similarity of centred GATv2 outputs for the same example under different queries and for different examples; **Context**: context share (%); ρ: Spearman's correlation between query similarity and same-example similarity (all $p < 10^{-100}$).

models	datasets	Repeated	Per ex.	Same	Diff	Context	ρ
Q 3-VL (2B)	HASOC *(T, 4)*	2523	4.8	0.62	0.01	38.7	0.42
	TOXICTAGS *(M, 4)*	2241	3.9	0.52	0.02	48.4	0.47
	MULTILINGUAL *(T, 2)*	8080	5.5	0.61	0.00	39.5	0.40
	MULTIMODAL *(M, 2)*	2971	5.4	0.65	0.00	34.8	0.52
I 3.5-VL (2B)	HASOC *(T, 4)*	2523	4.8	0.62	0.01	38.3	0.42
	TOXICTAGS *(M, 4)*	2241	3.9	0.45	0.01	55.3	0.58
	MULTILINGUAL *(T, 2)*	8080	5.5	0.60	0.00	40.4	0.39
	MULTIMODAL *(M, 2)*	2971	5.4	0.66	0.00	34.4	0.52



(a) QWEN 3-VL 2B



(b) INTERN 3.5-VL 2B

Figure 2: **Query conditioning on HASOC (T, 4):** Left: cosine similarity of centred GATv2 outputs for the same training example retrieved by different queries (blue) against different examples (orange). Right: the similarity of the same example's representations increases with the similarity of the two queries that retrieved it (mean per query-similarity decile with 95% confidence band).

same query generated by QWEN 3-VL-EMBEDDING-2B (Eq. 4.2), which acts as a baseline for the encoder's raw output. Both representations are L2-normalized. We evaluate (i) class clustering tightness using the cosine Silhouette score on mean-centered vectors, and (ii) linear decodability of label information via 5-fold stratified cross-validated logistic regression on standardized features. Crucially, neither metric relies on the model's ultimate predictions.

As detailed in Table 5, target classes cluster 5.7–14.2× more tightly within the context vectors than in the query embedding on HASOC, TOXICTAGS, and MULTIMODAL, consistent across both backbones and independent of k. This structural coherence is remarkable given that explicit labels

Table 5: **Label structure of the context vectors. Silhouette**: cosine Silhouette of class labels on mean-centred vectors, for the frozen query embedding z_q and for the context vectors $\bar{c}_q$ at 2/4/8 shots; **Gain**: 8-shot Silhouette relative to z_q. **Probe**: 5-fold linear-probe accuracy (%) on z_q and on the 8-shot $\bar{c}_q$; **Maj.**: majority-class accuracy.

models	datasets	Silhouette					Probe		
		z_q	2	4	8	Gain	z_q	$\bar{c}_q$	Maj.
Q 3-VL (2B)	HASOC *(T, 4)*	0.006	0.082	0.081	0.082	13.6×	64.8	61.2	54.1
	TOXICTAGS *(M, 4)*	0.003	0.048	0.049	0.048	13.8×	61.8	58.5	47.2
	MULTILINGUAL *(T, 2)*	0.006	0.010	0.010	0.010	1.8×	67.1	64.0	57.2
	MULTIMODAL *(M, 2)*	0.038	0.219	0.219	0.218	5.8×	72.9	69.3	65.3
I 3.5-VL (2B)	HASOC *(T, 4)*	0.006	0.086	0.085	0.086	14.2×	64.8	61.2	54.1
	TOXICTAGS *(M, 4)*	0.003	0.027	0.027	0.027	7.8×	61.8	58.2	47.2
	MULTILINGUAL *(T, 2)*	0.006	0.015	0.016	0.016	2.7×	67.1	64.2	57.2
	MULTIMODAL *(M, 2)*	0.038	0.216	0.216	0.215	5.7×	72.9	69.5	65.3

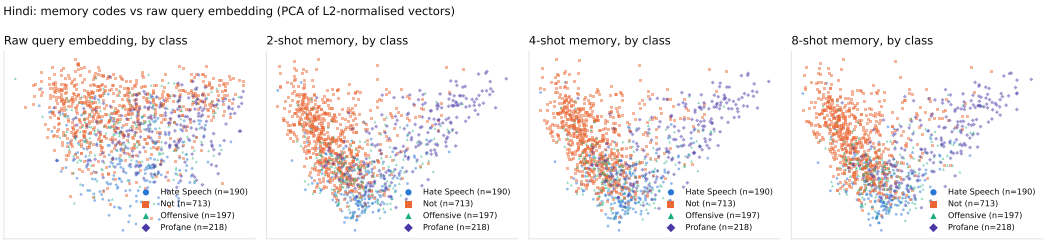
are absent during context building at inference; rather, this structure is deeply imprinted during training, where the SCL objective shapes the GATv2 space and the CE objective optimizes the context generator (Section 4.5). Interestingly, the linear probe accuracy is 2.8–3.6 points *lower* on $\bar{c}_q$ than on z_q (though still outperforming the majority class baseline). Because $\bar{c}_q$ is derived exclusively from the query's retrieved neighbors, it cannot inherently contain more direct information about the query than the original embedding does. Therefore, the context builder does not introduce new label information; instead, it *rearranges* the retrieved evidence to cleanly separate classes in the representation space. The PCA projections in Figures 3–5 (which natively limit artificial non-linear separation) visually corroborate this. For QWEN 3-VL 2B, the frozen embeddings on HASOC form a monolithic mixed cloud, whereas the context vectors distinctly isolate *not* and *profane* on opposite arms, with *hate speech* at the vertex and *Offensive* remaining mixed (Figure 3). On TOXICTAGS, *hateful* and *dangerous* diverge to the extremes, leaving *normal* and *offensive* central (Figure 4). On MULTIMODAL, context vectors successfully segregate the two source datasets (language Silhouette increases from 0.14 to 0.65 for both backbones), and within the BENGALI cluster, *normal* and *toxic* migrate to opposite poles (Figure 5). Because toxicity rates differ significantly between the sources (85% vs. 37%), a portion of the class Silhouette gain on MULTIMODAL is inherently tied to this source separation.

MULTILINGUAL presents a notable exception: its context vectors display negligible class organization (Silhouette ≤ 0.016; Figure 6), despite yielding substantial performance gains ($+23.86$ and $+27.56$ Macro-F1), as shown in Section D.4. This indicates that the extracted utility extends beyond mere label clustering. A plausible hypothesis is that the binary *toxic* label aggregates disparate policy definitions across HATEXPLAIN, AMHARIC, and HASOC, preventing a unified class-level geometry across the suite; conversely, language identity remains nearly perfectly decodable from the context vectors ($\geq 98\%$ probe accuracy).

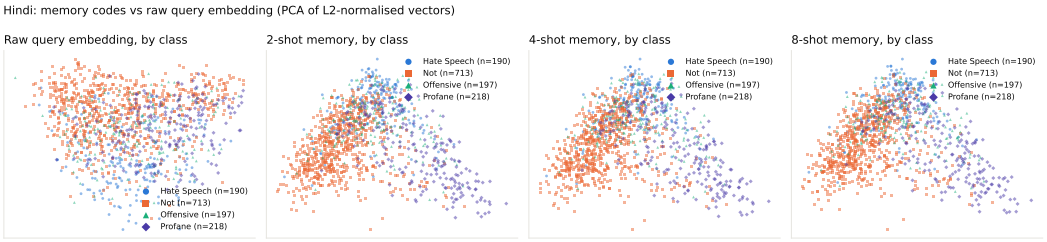
D.4 DOES THE VLM RELY ON THE CONTEXT OR ON THE EXTRA RESIDUAL?

The performance gains achieved by HOLOGRAM could theoretically arise simply from the introduction of additional trainable parameters (the extra residual layer) rather than the semantic content of the retrieved examples. To decouple these factors, we regenerate every test prediction at $k{=}8$ (the maximum support size, providing the strongest context signal) under three controlled conditions sharing an identical zero-shot prompt and identical trained modules: (i) *no context*, completely omitting context injection to recover the frozen VLM baseline; (ii) *random context*, replacing each selected GATv2 representation h_j with a Gaussian vector matching the mean and standard deviation across dimensions (preserving the norm), averaged over three distinct seeds; and (iii) the standard HOLOGRAM *context*. Condition (ii) effectively preserves the scale of the steering signal and the computational capacity of the context-to-memory pathway, isolating the impact of the retrieved semantic content. Predictions are extracted using a uniform parser across all conditions.

Table 6 and Figure 7 show that random content merely perturbs the VLM's baseline (yielding $+0.10$ to $+7.31$ Macro-F1, with seed-to-seed standard deviations ≤ 1.04). In stark contrast, the true HOLOGRAM context yields dramatic improvements of $+14.27$ to $+40.69$, maintaining a performance gap of at least 13.8 points across every configuration. This effect remains robust across individual languages within the combined suites (Table 7), notably driving massive improvements in low-

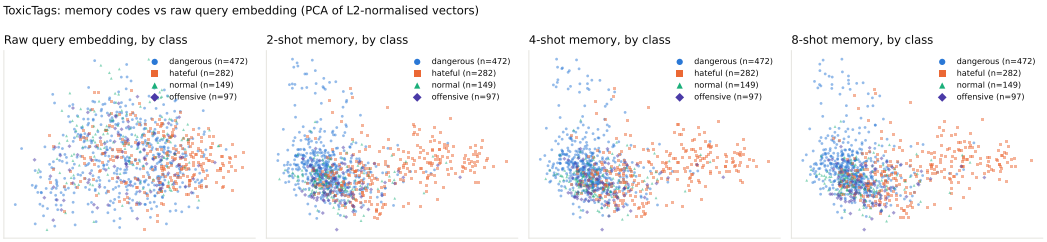


(a) QWEN 3-VL 2B

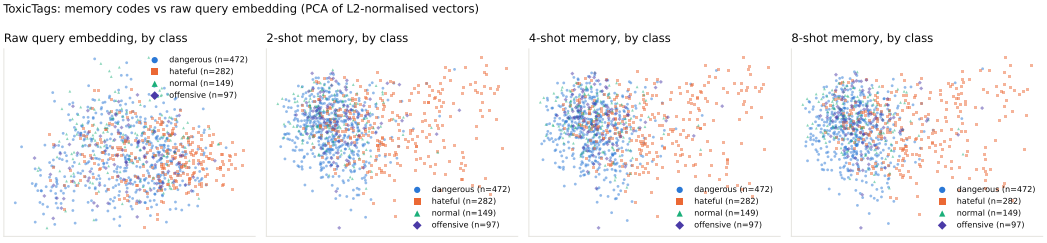


(b) INTERN 3.5-VL 2B

Figure 3: PCA of the frozen query embedding z_q (first column) and the context vectors $\bar{c}_q$ at 2/4/8 shots on HASOC (T, 4), colored by gold label.



(a) QWEN 3-VL 2B

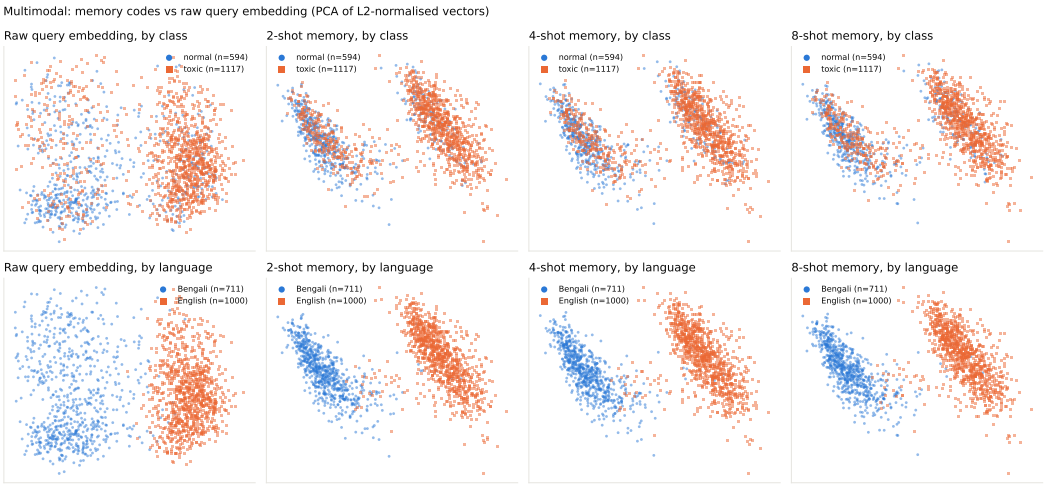


(b) INTERN 3.5-VL 2B

Figure 4: PCA of z_q and $\bar{c}_q$ on TOXICTAGS (M, 4), coloured by gold label.



(a) QWEN 3-VL 2B



(b) INTERN 3.5-VL 2B

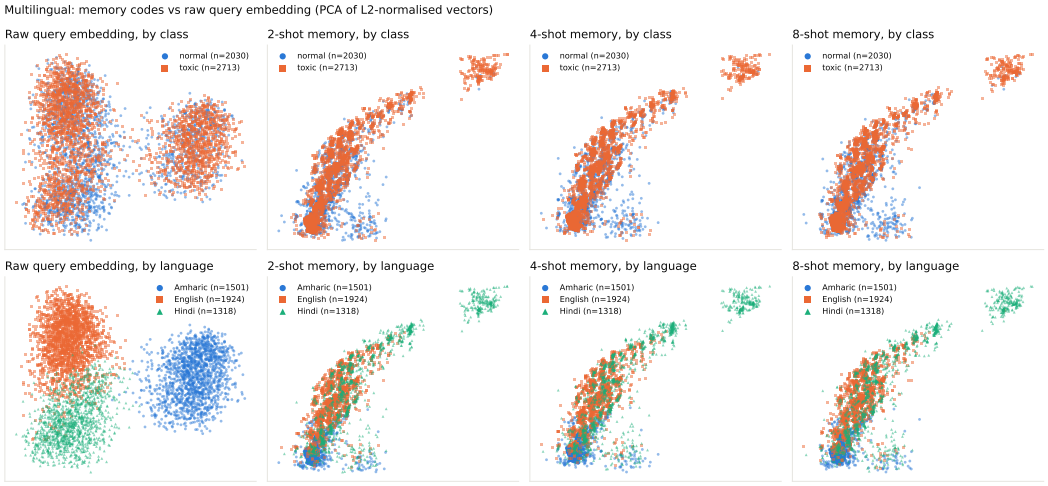
Figure 5: PCA of z_q and $\bar{c}_q$ on MULTIMODAL (M, 2), coloured by gold label (upper row of each panel) and by language (lower row; BENGALI = BHM, ENGLISH = TOXICTAGS).

Table 6: **Content versus presence of the context (Macro-F1, 8 shots) – Random**: mean ± standard deviation over three seeds. The last two columns give the change relative to no context. The HOLOGRAM values on HASOC (T, 4) and TOXICTAGS (M, 4) agree with the 8-shot values of Table 2 within 0.7 Macro-F1, and those on MULTILINGUAL and MULTIMODAL agree with Table 3 within 0.2; the residual differences stem from a corrected answer parser and run-to-run variation.

models	datasets	No context	Random	HOLOGRAM	ΔRandom	ΔHOLOGRAM
Q 3-VL (2B)	HASOC *(T, 4)*	35.51	35.61 ± 0.61	57.19	+0.10	**+21.68**
	TOXICTAGS *(M, 4)*	26.72	28.82 ± 0.22	55.52	+2.10	**+28.80**
	MULTILINGUAL *(T, 2)*	51.16	57.29 ± 0.48	75.02	+6.12	**+23.86**
	MULTIMODAL *(M, 2)*	57.59	58.02 ± 0.23	71.86	+0.44	**+14.27**
I 3.5-VL (2B)	HASOC *(T, 4)*	22.74	25.72 ± 1.04	57.81	+2.97	**+35.06**
	TOXICTAGS *(M, 4)*	22.25	22.35 ± 0.55	54.96	+0.10	**+32.72**
	MULTILINGUAL *(T, 2)*	49.54	56.38 ± 0.52	77.09	+6.85	**+27.56**
	MULTIMODAL *(M, 2)*	31.89	39.20 ± 0.64	72.58	+7.31	**+40.69**



(a) QWEN 3-VL 2B

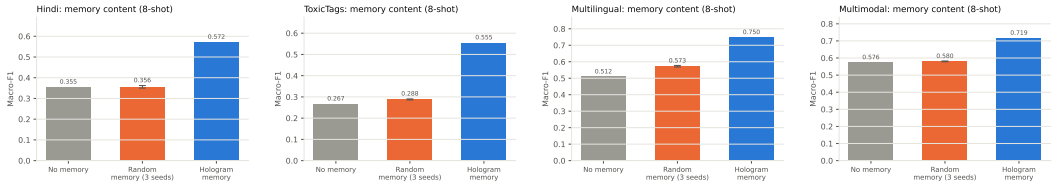


(b) INTERN 3.5-VL 2B

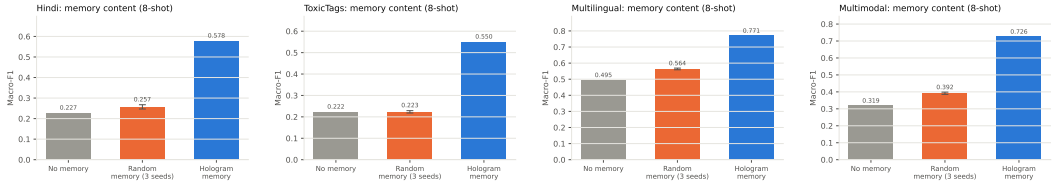
Figure 6: **PCA of z_q and $\bar{c}_q$ on MULTILINGUAL (T, 2)**, coloured by gold label (upper row of each panel) and by language (lower row).

Table 7: **Content versus presence of the context per language on the combined suites (Macro-F1, 8 shots) – Random**: mean over three seeds.

models	datasets	language	No context	Random	HOLOGRAM
Q 3-VL (2B)	MULTILINGUAL	AMHARIC	43.29	52.66	**68.09**
		ENGLISH	49.93	55.88	**74.04**
		HINDI	55.99	61.10	**81.50**
	MULTIMODAL	BENGALI	52.80	52.64	**59.93**
		ENGLISH	53.89	54.14	**62.38**
I 3.5-VL (2B)	MULTILINGUAL	AMHARIC	41.14	51.08	**70.90**
		ENGLISH	57.02	59.00	**77.37**
		HINDI	47.84	58.23	**81.22**
	MULTIMODAL	BENGALI	39.31	40.49	**65.02**
		ENGLISH	20.41	28.76	**59.91**



(a) QWEN 3-VL 2B



(b) INTERN 3.5-VL 2B

Figure 7: **Macro-F1 with no context, random context (mean ± std over three seeds), and the HOLOGRAM context** at 8 shots on HASOC (T, 4), TOXICTAGS (M, 4), MULTILINGUAL (T, 2), and MULTIMODAL (M, 2).

resource settings like AMHARIC (up to +29.76 over no context) and CODE-MIXED BENGALI memes. Therefore, HOLOGRAM's effectiveness is rooted in the semantic content extracted via the retrieval and GATV2-encoding pipeline, not merely the added parameter capacity. Together with Section D.3, this is consistent with the VLM leveraging the structured, label-organized evidence generated by the context builder. MULTILINGUAL, which achieves large gains despite weaker class clustering, implies that this semantic exploitation extends beyond simple class separation. We acknowledge that while random vectors match the mean and scale, they lack the specific dimensional distributions of genuine GATV2 outputs; nevertheless, they safely establish a baseline floor, never falling below the no-context performance on average.

D.5 HOW DOES THE CONTEXT ACT INSIDE THE VLM?

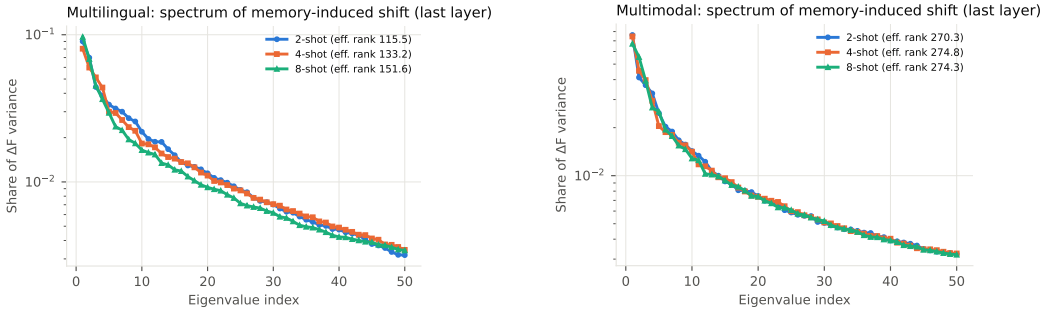
Adapting the spectral analysis methodology established by PMT (Wang et al., 2026), we rigorously characterize the context's representational impact at the final decoder layer. For each test prompt, we record the final layer's self-attention output both without context (F_b) and with the HOLOGRAM context at $k \in \{2, 4, 8\}$ (F_k). We then compute the shift $\Delta F = F_k - F_b \in \mathbb{R}^{n \times d_{\text{hidden}}}$ across all n tokens in the test suite, incorporating image tokens (0.97M tokens for MULTILINGUAL; 1.6M and 3.8M for MULTIMODAL with QWEN 3-VL 2B and INTERN 3.5-VL 2B, respectively). We perform an eigendecomposition of the covariance matrix $\Sigma = \frac{1}{n-1}(\Delta F - \mathbf{1}\mu^\top)^\top(\Delta F - \mathbf{1}\mu^\top) = V\Lambda V^\top$ (where μ denotes the mean shift over tokens) and report: the relative magnitude $\|\Delta F\|_F / \|F_b\|_F$ averaged across prompts; the variance captured by the top-10 and top-64 eigen-directions; and the effective rank $\exp(-\sum_i p_i \log p_i)$, utilizing $p_i = \lambda_i / \operatorname{tr}(\Lambda)$.

Because context injection occurs at *every* layer, the final-layer shift is an amalgamation of two phenomena: the final layer's localized injection and the compounded cascade of injections from prior layers propagating through the frozen network. The localized injection strictly resides in a mathematically bounded subspace: $\text{ELU}(q_{L,t}) \cdot M_L = \big(\text{ELU}(q_{L,t}) \cdot (\bar{c}_L \cdot A_L)\big) \cdot B_L$, which invariably falls within the r-dimensional row space of B_L. Defining $U \in \mathbb{R}^{d_{\text{hidden}} \times r}$ as an orthonormal basis for this space, the *direct share* $\operatorname{tr}(U^\top \Sigma U) / \operatorname{tr}(\Sigma)$ represents an upper bound on the proportion of the final-layer shift directly written at that layer (as it unavoidably includes propagated shifts that incidentally map to the same subspace). A random r-dimensional subspace captures $r/d_{\text{hidden}} = 64/2048 = 3.1\%$ in expectation. We also report the direct share based on the uncentered second moment $\frac{1}{n}\Delta F^\top \Delta F$ (incorporating the mean shift μ) and the proportional weight of the mean shift itself, $\|\mu\|^2 / \frac{1}{n}\|\Delta F\|_F^2$.

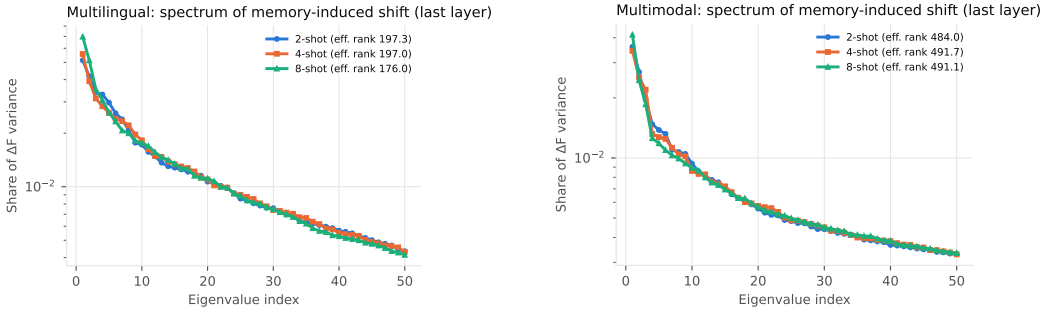
Table 8 and Figure 8 yield three critical insights. First, the context extensively alters the frozen model's computations, shifting the final-layer output by 21–67% of its norm. This shift scales monotonically with the number of shots (k) across all configurations, confirming the intended compounding effect

Table 8: **Representation effect of the HOLOGRAM context at the final decoder layer** ($d_{\text{hidden}} = 2048$, $r = 64$). **Shift**: mean $\|\Delta F\|_F / \|F_b\|_F$ (%); **Top-10 / Top-64**: variance of ΔF captured by the leading eigen-directions (%); **Eff. rank**: effective rank of Σ; **Direct**: share of ΔF lying in the row space of B_L (%), from the covariance and from the uncentred second moment, to be compared with 3.1% for a random subspace; **Mean**: share of the mean shift in the uncentred second moment (%).

models	datasets	k	Shift	Spectrum			Direct		Mean
				Top-10	Top-64	Eff. rank	centred	uncentred	
Q 3-VL (2B)	MULTILINGUAL	2	39.6	41.2	78.0	116	3.67	3.83	6.4
		4	50.4	38.5	75.0	133	3.76	4.03	8.0
		8	55.3	37.6	70.0	152	3.85	4.37	11.1
	MULTIMODAL	2	21.2	29.7	57.2	270	3.65	3.85	9.0
		4	32.0	29.2	56.6	275	3.67	3.90	11.7
		8	46.0	29.4	56.4	274	3.61	3.79	14.5
I 3.5-VL (2B)	MULTILINGUAL	2	40.6	29.5	67.6	197	3.30	3.29	2.2
		4	54.2	28.8	67.5	197	3.29	3.29	2.6
		8	67.3	31.3	69.3	176	3.31	3.31	4.0
	MULTIMODAL	2	36.6	16.6	40.1	484	3.17	3.17	1.5
		4	50.6	16.1	39.9	492	3.18	3.20	2.6
		8	65.9	15.8	39.7	491	3.19	3.22	3.8



(a) QWEN 3-VL 2B



(b) INTERN 3.5-VL 2B

Figure 8: **Eigen-spectrum of the final-layer representation effect** ΔF (top-50 eigenvalues normalised by the trace, log scale) at 2, 4, and 8 shots on MULTILINGUAL (left) and MULTIMODAL (right).

of summing multiple support contributions (Section 4.4). Second, this representational shift is highly structured yet avoids collapsing into a low-rank bottleneck: the top 64 directions capture 40–78% of the variance, maintaining an effective rank of 116–492 out of 2048, proving the steering signal is neither a singular overriding vector nor isotropic noise. Third, and most revealing of the underlying mechanism, the specific subspace directly written to by the final layer accounts for merely 3.2–3.9% of the centered variance (and 3.2–4.4% of the uncentered moment). This is negligibly higher than the 3.1% expected from a purely random subspace, whereas the theoretically optimal 64-dimensional subspace captures 40–78%. Because the direct share is an upper bound, we conclude that roughly $\sim$95% of the final-layer shift is the cumulative result of injections at earlier layers, organically carried forward and transformed by the frozen network's deep layers. The mean shift itself explains only 1.5–14.5% of the effect, indicating the context modulates distinct tokens heterogeneously rather than acting as a static offset. Ultimately, HOLOGRAM does not simplistically overwrite the VLM's output with an aggressive label signal; instead, its layer-wise, low-rank context intricately redirects the internal computational pathways of the frozen model, validating the core motivation for per-layer steering (Section 4.4).

D.6 SCOPE

While these aggregate analyses robustly validate each stage of the HOLOGRAM pipeline, they do not uniquely map the label organization of the context vector to the specific performance gain achieved on a per-query basis. Furthermore, our definition of *query-conditioned* encompasses the combination of the query itself alongside its retrieved neighbors, variables that these analyses do not isolate. Finally, all spectral and representational investigations reported here were conducted using the 2B backbone models.