On the Additive Compositionality of Task Vectors in Vision-Language Models

Yuting Shi and Houjing Wei and Naoya Inoue
Japan Advanced Institute of Science and Technology
{syting,houjing.w,naoya-i}@jaist.ac.jp

Abstract

In-context learning (ICL) in large language models (LLMs) has been shown to operate through task vectors—the representation that summarizes the mapping induced by in-context demonstrations and can be composed by simple arithmetic operations. While this phenomenon is well studied in LLMs, its extension to vision-language models (VLMs) remains underexplored. In this work, we systematically examine the additive compositionality of in-context task vectors in VLMs, extracted from text-side hidden representations. Specifically, we construct compositional visual reasoning tasks with clearly defined subtasks and extract task vectors from few-shot demonstrations. Empirical experiments show that the vector for a complex task can be approximated by adding the vectors of its constituent subtasks. Beyond this, we analyze token-level contextual embeddings and show that additive composition arises because complex-task representations emerge as the superposition of atomic subtask components.

1 Introduction

Recent work in natural language processing (NLP) has extensively examined the mechanism of ICL (Min et al., 2022; Olsson et al., 2022; Wei et al., 2022; Xie et al., 2022). Several studies show that the effect of an in-context prompt can be summarized as a task vector—an activation pattern capturing task-specific behavior, which can be extracted from few-shot demonstrations (Hendel et al., 2023; Todd et al., 2023). Such vectors in LLMs can often be combined arithmetically, e.g., by addition or subtraction, to induce new behaviors (Liu et al., 2023b).

However, while recent studies have begun extending task-vector analysis to multimodal settings (Luo et al., 2024; Huang et al., 2024a), the compositionality of task vectors—whether they can be additively combined to represent complex multimodal reasoning—remains largely unexplored.

Although VLMs inherit the decoder architecture of LLMs, their visual encoders and cross-modal alignment components add representational complexity. Hence, it remains unclear whether the task-vector arithmetic observed in text-only models can be extended to VLMs, since visual and textual spaces may not be perfectly aligned, and cross-modal interactions could disrupt the linearity of task-vector arithmetic.

To address this gap, we systematically investigate the additive compositionality of task vectors in VLMs. We design compositional visual reasoning tasks with clearly defined subtasks, extract task vectors from in-context demonstrations, and test whether the task vector for a complex task can be approximated by summing those of its atomic subtasks. We further analyze why compositionality holds through contextual embedding alignment.

Our main contributions are as follows: (1) We show that in-context task vectors in VLMs are additively compositional—complex-task vectors can be reconstructed from atomic ones, matching few-shot performance and exhibiting representational and causal alignment; (2) Our token-level contextual embedding analysis shows that complex-task representations preserve the structure of their atomic subtasks, providing representational evidence consistent with additive compositionality.

2 Related Work

2.1 In-context Task Vectors

Recent studies reveal that transformer activations encode reusable task representations during in-context learning. Hendel et al. (2023) show that such internal vectors capture task rules, while Todd et al. (2023) identify function vectors via targeted attention-head interventions. Luo et al. (2024) extend this to VLMs, finding modality-invariant task

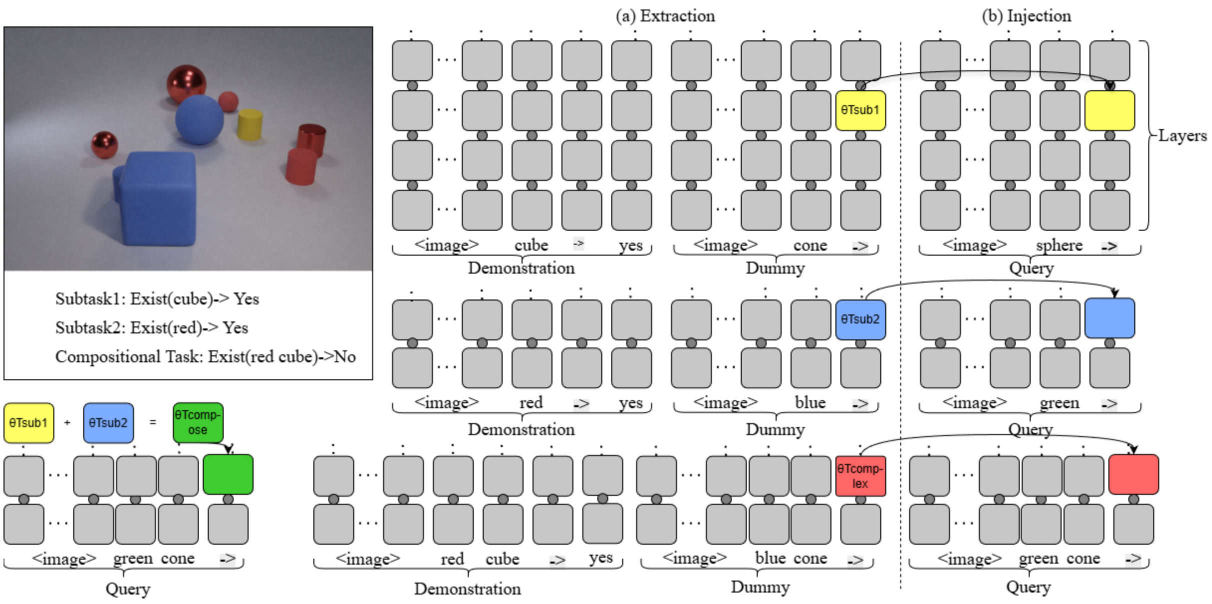


Figure 1: **Pipeline for task vector extraction, injection, and composition in VLMs.** (a) Extract subtask vectors from in-context demonstrations paired with a fixed dummy query (Extraction Prompt), reading the last text token representation. (b) Inject task vectors into the Injection Target Prompt (single real query, no demonstrations) to steer model outputs.

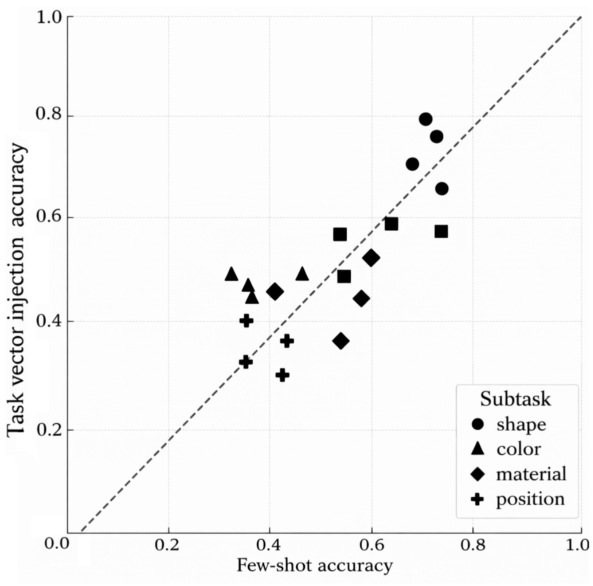


Figure 2: **Subtask-level injection vs. 3-shot performance comparison.** Points near the dashed $y{=}x$ line indicate that task vectors extracted from the same demonstrations and injected at inference can reproduce 3-shot performance without demonstrations.

vectors shared across inputs. Unlike prior work on task arithmetic in parameter space (Ilharco et al., 2023), we focus on in-context task vectors that emerge in activation space.

2.2 Compositionality in VLMs

Instruction-tuned VLMs such as LLaVA-1.5 and InstructBLIP (Liu et al., 2023a; Dai et al., 2023) perform well in CLEVR, GQA, NLVR2 (Johnson et al., 2017; Hudson and Manning, 2019; Suhr et al., 2019), yet they still struggle with compositional generalization (Huang et al., 2024b). We focus on the inference stage and test whether such compositional abilities can emerge implicitly through additive composition of in-context task vectors. Related work has analyzed visual encoders (Berasi et al., 2025) or linearly combined trained soft prompts (Perera et al., 2023).

3 Method

We hypothesize that task vectors in VLMs exhibit *additive compositionality*—that is, the vector for a complex reasoning task can be approximated by the sum of its constituent subtask vectors.

We employ three prompt formats to operationalize task-vector extraction and evaluation: (i) a **Few-Shot Prompt** (k demonstrations + query) used to elicit task-specific behavior; (ii) an **Extraction Prompt** (same demonstrations + fixed dummy query) used to extract task vectors; and (iii) an **Injection Prompt** (query only) used to test the injected vectors during inference. Task vectors are defined as the last-token hidden state at layer ℓ

Dataset / Task	3-shot (IDEFICS-8B)	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	Random	Injection layer
CLEVR (10 two-attribute combinations)					
color-material	0.92 ± 0.015	0.72 ± 0.040	0.90 ± 0.010	0.22 ± 0.031	17
color-size	0.76 ± 0.274	0.85 ± 0.123	0.63 ± 0.235	0.44 ± 0.052	17
shape-color	0.83 ± 0.263	0.97 ± 0.025	0.76 ± 0.344	0.41 ± 0.023	17
shape-position	0.80 ± 0.061	0.69 ± 0.389	0.67 ± 0.155	0.39 ± 0.067	20
size-position	0.69 ± 0.240	0.44 ± 0.206	0.56 ± 0.316	0.37 ± 0.054	20
shape-material	0.76 ± 0.143	0.70 ± 0.111	0.71 ± 0.092	0.40 ± 0.065	17
shape-size	0.68 ± 0.175	0.74 ± 0.097	0.67 ± 0.142	0.42 ± 0.073	17
size-material	0.59 ± 0.166	0.51 ± 0.132	0.49 ± 0.118	0.38 ± 0.059	20
position-material	0.65 ± 0.152	0.60 ± 0.121	0.61 ± 0.109	0.40 ± 0.063	20
color-position	0.67 ± 0.178	0.63 ± 0.139	0.65 ± 0.124	0.43 ± 0.071	17
GQA (3 two-attribute combinations)					
shape-color	0.74 ± 0.145	0.65 ± 0.097	0.69 ± 0.101	0.21 ± 0.054	17
shape-material	0.71 ± 0.139	0.62 ± 0.105	0.64 ± 0.096	0.37 ± 0.052	17
color-material	0.69 ± 0.132	0.60 ± 0.098	0.62 ± 0.092	0.21 ± 0.049	17

Table 1: Task vector composition results on CLEVR (10 two-attribute combinations) and GQA (3 two-attribute combinations) datasets using the IDEFICS-8B model. $\theta_{T_{\text{complex}}}$ denotes vectors extracted from compositional task prompts; $\theta_{T_{\text{compose}}}$ denotes the composed sum of atomic subtask vectors. Random denotes a random composition baseline with randomly composed subtask vectors injection. The last column shows the best injection layer for each task.

under the Extraction Prompt as follows:

$$\theta^{(\ell)} = h^{(\ell)}_{\text{Extraction}}$$

The task vector is injected into the same layer of the query prompt as follows:

$$\tilde{h}^{(\ell)} = h^{(\ell)}_{\text{Query}} + \theta^{(\ell)}.$$

Using these prompts, we implement a three-step framework (Fig. 1): (1) Construct structured visual reasoning tasks with defined atomic and composite queries; (2) Extract task vectors from demonstrations using the Extraction Prompt; and (3) Inject composed vectors into query-only prompts to evaluate additive compositionality by comparing their effects with directly extracted complex-task vectors. A compositional task T_{complex} is formed by combining atomic subtasks T_{sub} that probe single attributes. For two subtasks T_{sub1} and T_{sub2}, their composed vector is as follows:

$$\theta^{(\ell)}_{T_{\text{compose}}} = \theta^{(\ell)}_{T_{\text{sub1}}} + \theta^{(\ell)}_{T_{\text{sub2}}},$$

which is injected into the corresponding complex query and compared against $\theta^{(\ell)}_{T_{\text{complex}}}$ to assess additive compositionality. This comparison is used solely to assess representational equivalence between task vectors.

4 Experiments

4.1 Setup

We evaluate the additive compositionality of in-context task vectors across atomic and compositional visual reasoning tasks on CLEVR and GQA (Laurençon et al., 2023a; Liu et al., 2023a; Johnson et al., 2017; Hudson and Manning, 2019). In CLEVR, we define five atomic subtasks—shape, color, material, size, and position—yielding ten pairwise compositional tasks (e.g., shape–color, color–material). In GQA, we use three atomic subtasks (shape, color, material) and their three possible pairwise combinations.

We use three representative VLMs: IDEFICS-8B, LLaVA-1.5-7B, and Qwen2-VL-7B-Instruct (Laurençon et al., 2023b; Liu et al., 2023a; Wang et al., 2024). All models are evaluated on the same 200 test instances per task, and each result is averaged over three random seeds.

4.2 Results

Task Vector Injection Reproduces Subtask Performance. We first evaluate whether task vectors can recover task-specific behavior without demonstrations. As shown in Fig. 2, their injection performance closely matches the few-shot results across all subtasks and models, indicating that task vectors effectively capture and transfer subtask knowledge (see Appendix A for detailed results on GQA and other models).

Task Vectors are Additively Compositional. We test whether a compositional task vector $\theta_{T_{\text{complex}}}$ can be approximated by summing its atomic subtasks as defined in Section 3. Also, we randomly sum two subtask vectors from attibute pool to make

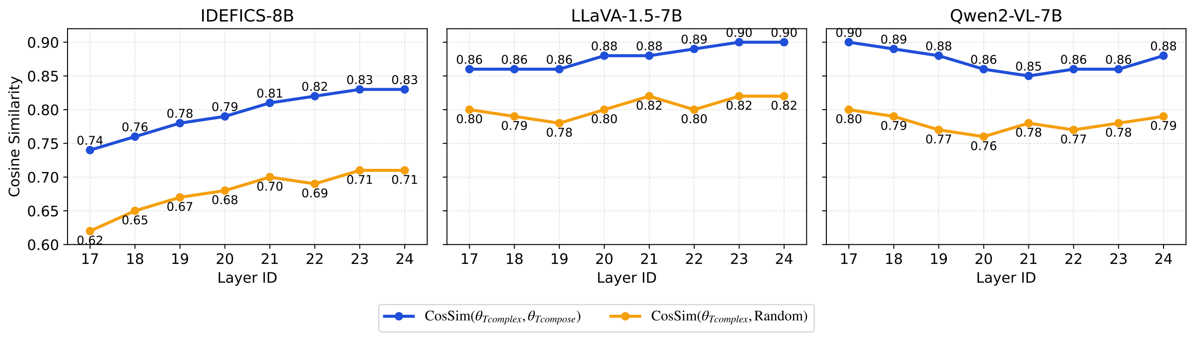


Figure 3: **Layerwise cosine similarity between composed and combined task vectors.** Representative case: CLEVR dataset, IDEFICS-8B, $k = 3$. $\theta_{T_{\text{compose}}}$ consistently shows higher similarity to $\theta_{T_{\text{complex}}}$ than the random-composition baseline across middle layers.

random composition as the baseline and inject it into Injection Prompt. As shown in Table 1, injecting $\theta_{T_{\text{compose}}}$ achieves accuracy comparable to directly using $\theta_{T_{\text{complex}}}$, while random composition perform notably worse. The cosine similarity in Fig. 3 further shows that $\theta_{T_{\text{complex}}}$ aligns more closely with $\theta_{T_{\text{compose}}}$ than with random composition, confirming that compositional behavior emerges from the linear recombination of subtask representations. We also find that IDEFICS-8B shows the strongest additive approximation (e.g., shape–color: $\theta_{T_{\text{complex}}} = 0.68$ vs. $\theta_{T_{\text{compose}}} = 0.71$ in Table 5), Qwen2-VL-7B exhibits moderate but stable approximation (e.g., 0.73 vs. 0.50), while LLaVA-1.5-7B shows the weakest alignment. Importantly, these differences closely mirror the models' zero-shot and few-shot capabilities (Table 5 and 6 in Appendix A).

To further stress-test additive compositionality beyond pairwise settings, we evaluate three-attribute compositional tasks. Specifically, we construct four compositional tasks of three-attributes by combining atomic subtasks from shape, color, size, material, position. The results are summarized in Table 2. Overall, we observe that additive composition remains effective in several tri-attribute tasks, significantly outperforming random baselines and in some cases even matching or exceeding directly extracted complex-task vectors.

For tasks involving only object-centric attributes—such as color-size-material and shape-color-size: $\theta_{T_{\text{compose}}}$ achieves strong performance (0.96 and 0.90, respectively), closely matching or surpassing $\theta_{T_{\text{complex}}}$. This suggests that atomic attribute representations can be linearly superposed even when more than two attributes are involved, supporting the scalability of additive compositionality beyond pairwise settings.

Task	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	**Random**
color-size-material	0.86	0.96	0.58
shape-color-size	0.87	0.90	0.56
shape-color-position	0.77	0.52	0.37
shape-material-position	0.57	0.62	0.43

Table 2: Results on tri-attribute compositional tasks.

5 Analysis

5.1 Importance of Attention Heads

We argue that if $\theta_{T_{\text{compose}}}$ and $\theta_{T_{\text{complex}}}$ rely on similar internal components to affect model predictions, then overlap in the attention heads implicated by task-vector injection provides complementary evidence—beyond behavioral accuracy—that they encode similar task representations.

To identify attention heads that are important for task-vector-induced behavior, we measure the change in model predictions after ablating each head's output (Todd et al., 2023). For each attention head $h_{l,i}$, we quantify the effect of ablating the head under a given condition as follows:

$$\text{CIE}(h_{l,i}) = \|\text{logits} - \text{logits}_{\text{zeroed}}\|_2.$$

We compute this quantity under $\theta_{T_{\text{compose}}}$ injection, $\theta_{T_{\text{complex}}}$ injection, and a zero-shot baseline, and define

$$\Delta\text{CIE}(h_{l,i}) = \left|\text{CIE}_{\text{injection}} - \text{CIE}_{\text{zero-shot}}\right|$$

to isolate heads whose influence changes most strongly due to task-vector injection.

We then rank all heads by ΔCIE and compare the top-10 head sets between complex-task injection and composed-vector injection. We observe

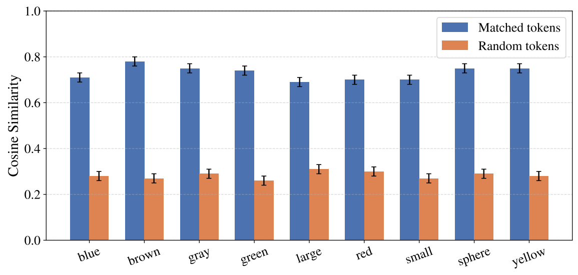


Figure 4: **Token-level embedding alignment for compositional tasks (color–size).** Average cosine similarity between identical tokens in few-shot atomic and complex prompts. Confirming that each subtask contributes an independent feature subspace that can be linearly combined in compositional task.

an overlap of 0.7 suggestting that both types of vectors engage largely overlapping sets of functionally important heads, supporting their representational similarity.

5.2 Similarity of Contextual Representations

We hypothesize that the additive compositionality of task vectors is reflected directly in the model's contextual representations: the representation of a complex task can be approximated by the linear combination of constituent subtask representations.

To verify this, we analyze token-level contextual embeddings in the representative **color-size** task group on IDEFICS-8B. We focus on the same tokens but in different context that carry identical semantics in complex and atomic task prompt types (e.g., the token cube appearing in "<image> cube $\rightarrow$ yes" and "<image> red cube $\rightarrow$ no"). We then compute cosine similarities between paired tokens and average the results across examples. We additionally compute cosine similarity between mismatched tokens as random baseline— for example, taking the token `brown` from an atomic color prompt and comparing it to a different token from the corresponding complex task.

As shown in Fig. 4, matched tokens in the color–size task reach cosine similarities near 0.9, while random pairs stay around 0.3. The nearly identical embeddings suggest that complex tasks are encoded as superpositions of atomic components, with each subtask contributing an independent feature subspace. Task vectors from atomic prompts can thus be additively combined to reconstruct the complex-task representation.

6 Conclusion

We provide the first systematic evidence that in-context task representations in VLMs follow the principle of additive compositionality. Task vectors extracted from few-shot demonstrations can be additively combined to reconstruct complex-task behavior, achieving few-shot–level performance without demonstrations. Analysis of token-level embeddings reveals that this compositionality stems from the superposition of atomic subtask representations within the model's hidden space, offering an explanation for why linear task-vector arithmetic yields modular and interpretable reasoning.

Limitations

Our study focuses on controlled visual reasoning tasks (CLEVR, GQA) with minimal prompts, which may not reflect open-domain multimodal reasoning. We analyze only direct vector addition, leaving learned compositions for future work.

Acknowledgement

This work is partially supported by the Nakajima Foundation.

References

Davide Berasi, Matteo Farina, Massimiliano Mancini, Elisa Ricci, and Nicola Strisciuglio. 2025. Not only text: Exploring compositionality of visual representations in vision-language models. *arXiv preprint arXiv:2503.17142*.

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

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*.

Brandon Huang, Chancharik Mitra, Assaf Arbelle, Leonid Karlinsky, Trevor Darrell, and Roei Herzig. 2024a. Multimodal task vectors enable many-shot multimodal in-context learning. *Preprint*, arXiv:2406.15334. Published in NeurIPS 2024.

Irene Huang, Wei Lin, Muhammad Jehanzeb Mirza, Jacob Hansen, Sivan Doveh, Victor Butoi, Roei Herzig, Assaf Arbelle, Hilde Kuehne, Trevor Darrell, and 1 others. 2024b. Conme: Rethinking evaluation of compositional reasoning for modern vlms. *Advances in Neural Information Processing Systems*, 37:22927–22946.

Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR) 2023*. ArXiv preprint arXiv:2212.04089.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M. Rush, Douwe Kiela, Matthieu Cord, and Victor Sanh. 2023a. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Preprint*, arXiv:2306.16527.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M. Rush, Douwe Kiela, Matthieu Cord, and Victor Sanh. 2023b. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Preprint*, arXiv:2306.16527.

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

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. 2023b. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*.

Grace Luo, Trevor Darrell, and Amir Bar. 2024. Vision-language models create cross-modal task representations. *arXiv preprint arXiv:2410.22330*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *Preprint*, arXiv:2202.12837.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

Pramuditha Perera, Matthew Trager, Luca Zancato, Alessandro Achille, and Stefano Soatto. 2023. Prompt algebra for task composition. *arXiv preprint arXiv:2306.00310*.

Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. A corpus for reasoning about natural language grounded in photographs. *Preprint*, arXiv:1811.00491.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Preprint*, arXiv:2206.07682.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. *Preprint*, arXiv:2111.02080.

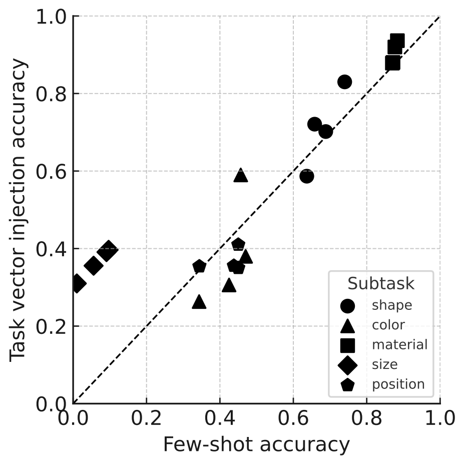
A Appendix

Hyperparameter Configuration. We use the default decoding and model configuration provided by the corresponding model implementations, with the following settings made explicit for reproducibility. decoding is performed with `max_new_tokens = 2` and `do_sample = False`, corresponding to deterministic greedy decoding without temperature sampling. We set `use_cache = False` to ensure consistent extraction of hidden states, and enable `return_hidden_states = True` when analyzing layer-wise contextual embeddings. All inputs are padded to equal length (`padding = True`) for batched inference. Decoding is therefore fully deterministic, ensuring that observed performance differences stem from task-vector manipulation rather than stochastic variation.

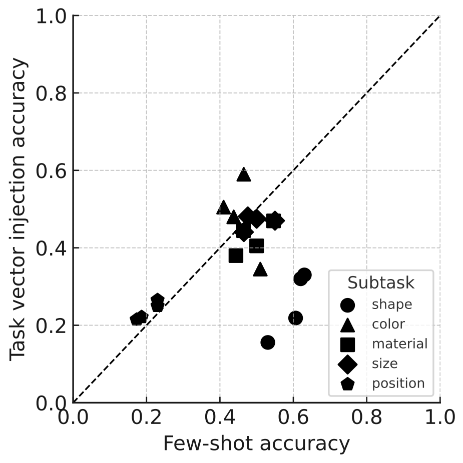
Models. LLaVA-1.5 (Liu et al., 2023a) is an open-source, transformer-based chatbot that learns to follow multimodal instructions. It achieves this by taking a foundational LLaMA or Vicuna language model and further training it with instruction-following data created by GPT. IDEFICS-8B (Laurençon et al., 2023b) an open-source, multimodal AI model capable of processing any mix of image and text inputs to generate text outputs. Qwen2 (Wang et al., 2024) series comprises multiple sizes of decoder language models, offered as foundational models and specialized chat versions. These models leverage the Transformer architecture and are distinguished by advanced components such as SwiGLU activation, attention QKV bias, and Group Query Attention. Furthermore, Qwen2 features an improved tokenizer designed for adaptability across numerous natural languages and programming codes.

Reproducibility and Code Release We will release the code upon publication at: `https://github.com/SSSSSeki/task-vector-compositionality`

AI Assistance Disclosure. The authors acknowledge the use of an AI writing assistant for refining sentence structure, improving grammatical accuracy, and enhancing the clarity of the manuscript's language. All scientific content and interpretations remain the sole responsibility of the authors.



(a) LLaVA-1.5-7B



(b) Qwen2-VL-7B-Instruct

Figure 5: Subtask-level injection vs. 3-shot performance comparison on LLaVA-1.5-7B and Qwen2-VL-7B-Instruct.

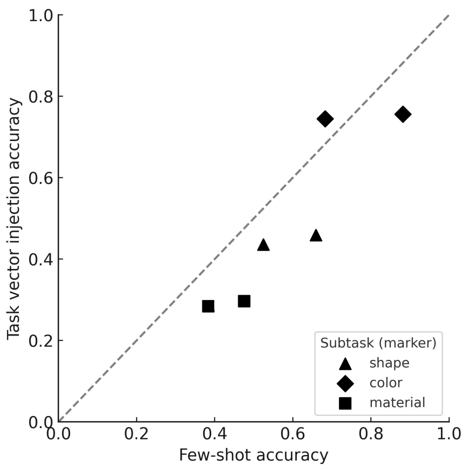


Figure 6: Subtask-level injection vs. 3-shot performance comparison on GQA.

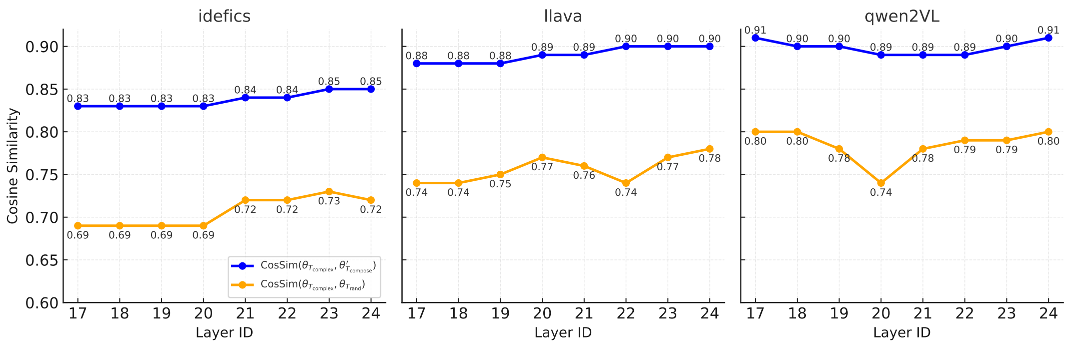


Figure 7: **Layerwise cosine similarity between composed and combined task vectors.** Representative case: GQA dataset, IDEFICS-8B, $k = 3$.

Dataset / Task	3-shot	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	Injection layer
CLEVR (5 two-attribute combinations)				
color_material	0.60 ± 0.12	0.11 ± 0.19	0.17 ± 0.18	17
color_size	0.37 ± 0.02	0.12 ± 0.11	0.14 ± 0.12	17
shape_color	0.70 ± 0.04	0.10 ± 0.17	0.18 ± 0.16	17
shape_position	0.81 ± 0.19	0.81 ± 0.00	0.22 ± 0.09	17
size_position	0.95 ± 0.13	0.95 ± 0.00	0.27 ± 0.10	18
GQA (3 two-attribute combinations)				
shape_color	0.71 ± 0.14	0.65 ± 0.10	0.69 ± 0.08	17
shape_material	0.68 ± 0.13	0.61 ± 0.09	0.64 ± 0.10	17
color_material	0.64 ± 0.13	0.58 ± 0.10	0.61 ± 0.09	17

Table 3: Task vector composition results on CLEVR (5 two-attribute combinations) and GQA (3 two-attribute combinations) using the **LLaVA-1.5-7B** model.

Dataset / Task	3-shot	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	Injection layer
CLEVR (5 two-attribute combinations)				
color_material	0.44 ± 0.11	0.68 ± 0.01	0.19 ± 0.11	21
color_size	0.65 ± 0.04	0.69 ± 0.07	0.22 ± 0.08	24
shape_color	0.34 ± 0.08	0.35 ± 0.08	0.18 ± 0.10	21
shape_position	0.56 ± 0.03	0.59 ± 0.12	0.23 ± 0.09	22
size_position	0.79 ± 0.18	0.90 ± 0.08	0.25 ± 0.11	22
GQA (3 two-attribute combinations)				
shape_color	0.69 ± 0.15	0.66 ± 0.09	0.62 ± 0.10	22
shape_material	0.65 ± 0.13	0.63 ± 0.10	0.59 ± 0.11	22
color_material	0.63 ± 0.13	0.61 ± 0.10	0.58 ± 0.11	22

Table 4: Task vector composition results on CLEVR and GQA using the **Qwen2-VL-7B-Instruct** model.

Model	**Shots**	**Zeroshot eval**	**Few-shot eval**	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	**Best layer**
			Shape–Material			
IDEFICS-8B	1-shot	0.00	0.72	0.41 ± 0.05	0.38 ± 0.04	18
IDEFICS-8B	2-shot	0.00	0.72	0.77 ± 0.04	0.68 ± 0.05	19
Qwen2-VL-7B-Instruct	1-shot	0.00	0.60	0.26 ± 0.06	0.20 ± 0.05	22
Qwen2-VL-7B-Instruct	2-shot	0.00	0.63	0.53 ± 0.06	0.47 ± 0.05	22
LLaVA-1.5-7B	1-shot	0.00	0.45	0.25 ± 0.05	0.27 ± 0.05	21
LLaVA-1.5-7B	2-shot	0.00	0.45	0.33 ± 0.05	0.30 ± 0.05	21
			Shape–Color			
IDEFICS-8B	1-shot	0.18	0.77	0.32 ± 0.03	0.35 ± 0.03	16
IDEFICS-8B	2-shot	0.18	0.83	0.68 ± 0.04	0.71 ± 0.04	21
Qwen2-VL-7B-Instruct	1-shot	0.00	0.72	0.48 ± 0.09	0.22 ± 0.05	23
Qwen2-VL-7B-Instruct	2-shot	0.00	0.75	0.73 ± 0.09	0.50 ± 0.07	23
LLaVA-1.5-7B	1-shot	0.00	0.70	0.10	0.25 ± 0.04	21
LLaVA-1.5-7B	2-shot	0.00	0.73	0.18	0.28 ± 0.05	24

Table 5: Few-shot scaling analysis (0–2 shots) of task vector composition on representative **CLEVR** tasks. Results are shown for **shape–material** (top) and **shape–color** (bottom) combinations. Only these tasks are evaluated for 0–2 shots to complement the main 3-shot experiments.

Model	**Shots**	**Zeroshot eval**	**Few-shot eval**	$\theta_{T_{\text{complex}}}$	$\theta_{T_{\text{compose}}}$	**Best layer**
			Shape–Material			
IDEFICS-8B	1-shot	0.00	0.70	0.33 ± 0.05	0.38 ± 0.05	19
IDEFICS-8B	2-shot	0.00	0.70	0.79 ± 0.04	0.70 ± 0.04	20
Qwen2-VL-7B-Instruct	1-shot	0.00	0.56	0.21 ± 0.06	0.18 ± 0.05	22
Qwen2-VL-7B-Instruct	2-shot	0.00	0.60	0.54 ± 0.06	0.45 ± 0.05	22
LLaVA-1.5-7B	1-shot	0.00	0.10	0.21 ± 0.05	0.23 ± 0.05	21
LLaVA-1.5-7B	2-shot	0.00	0.50	0.21 ± 0.05	0.25 ± 0.05	21
			Shape–Color			
IDEFICS-8B	1-shot	0.21	0.75	0.03 ± 0.02	0.18 ± 0.02	16
IDEFICS-8B	2-shot	0.21	0.83	0.70 ± 0.05	0.76 ± 0.03	21
Qwen2-VL-7B-Instruct	1-shot	0.00	0.75	0.55 ± 0.10	0.17 ± 0.05	24
Qwen2-VL-7B-Instruct	2-shot	0.00	0.73	0.80 ± 0.09	0.53 ± 0.07	23
LLaVA-1.5-7B	1-shot	0.00	0.75	0.00	0.27 ± 0.04	21
LLaVA-1.5-7B	2-shot	0.00	0.75	0.01	0.24 ± 0.05	24

Table 6: Few-shot scaling analysis (0–2 shots) of task vector composition on representative **GQA** tasks. Results are shown for **shape–material** (top) and **shape–color** (bottom) combinations. Only these tasks are evaluated for 0–2 shots to complement the main 3-shot experiments.