

NanoVQA

Stanford CS224N Custom Project

Ellen Xu

Department of Computer Science
Stanford University
ellenjxu@stanford.edu

Abstract

We present **NanoVQA**, a minimal vision-language model pairing a frozen CLIP-ViT encoder with a small MLP adapter and a GPT-2 decoder (124M parameters, $\sim 300\times$ smaller than LLaVA). Trained on VQAv2 [1] for under six hours on a single A10G GPU, it achieves **60.09%** test-standard accuracy at 3.40×10^{17} training FLOPs—on the Pareto frontier of accuracy vs. compute among published VLMs. A scaling analysis across decoder size, adapter bottleneck, CLIP token count, and training data shows that data scale is a dominant factor consistent with log-linear improvement; future experimentation is needed to confirm model scaling results.

1 Key Information to include

- Mentor: Sarah Chen
- External Collaborators (if you have any): None
- Sharing project: No
- Late Days (if you are using any):

Team member	Late days remaining
Ellen Xu	3
Total pooled	3
Team size	1
Late days applied for the report	3

2 Introduction

Humans understand the world through multi-modal channels spanning vision and language. To develop general-purpose assistants (e.g. robotics, computer-use agents), therefore requires models that can jointly process images and text. Early attempts kept vision and language largely separate: CNNs or ViTs were fine-tuned for single tasks such as image captioning, while LLMs were instruction-tuned exclusively on text.

Visual instruction tuning bridged this gap by extending instruction tuning to vision-and-text inputs [2]. Models such as LLaVA, Flamingo-80B, and InstructBLIP have pushed VQAv2 accuracy to over 81%. The current state of the art, Molmo2-8B, reaches 87.07%. However, all of these models require billions of parameters and petaflops of training compute, making them unsuitable for edge devices, fast data-labeling pipelines, or experimentation.

This paper asks: *how compute-efficient can visual instruction tuning be?* We study this question through a minimal, reproducible setup using NanoGPT/GPT-2 (124M parameters) as the language decoder, $\sim 300\times$ smaller than Vicuna-13B in LLaVA. Since the CLIP encoder is frozen, only a small MLP adapter (787K parameters) and the GPT-2 decoder require fine-tuning, reducing compute costs.

Beyond the main result, we investigate several additional research questions: (1) How does performance scale with decoder size? (2) How does the adapter bottleneck dimension (and architecture) affect accuracy? (3) How does the number of CLIP spatial tokens affect accuracy? (4) What is the data-scaling behaviour?

Our model, **NanoVQA**, was trained on the VQAv2 benchmark [1] and achieved **60.09%** accuracy on the test-standard split using only $\approx 3.40 \times 10^{17}$ training FLOPs—three to four orders of magnitude fewer than well-known published models. The model trains end-to-end in under six hours on a single NVIDIA A10G GPU.

To summarize, the main contributions:

- A minimal NanoVQA architecture built on a frozen CLIP encoder, MLP adapter, and a GPT-2 decoder, trained end-to-end on VQAv2.
- Fine-tuned on 443,757 VQAv2 training samples, achieving 60.09% test-standard accuracy on the VQAv2 benchmark—while using 3–4 orders of magnitude fewer training FLOPs and sitting on the Pareto frontier of accuracy vs. compute.
- A scaling analysis across decoder size, adapter architecture, CLIP token count, and training set size.

The code is open-sourced at <https://github.com/xuellen/nanovqa>. Model checkpoints and cached CLIP features are available on HuggingFace at <https://huggingface.co/ellen2imagine/nanovqa>.

3 Related Work

Visual instruction tuning. LLaVA [2] described the recipe for visual instruction tuning: a frozen CLIP-ViT vision encoder, a learned linear projection adapter, and a fine-tuned LLM decoder (Vicuna-13B). The model is trained on 80K image-instruction pairs generated by GPT-4 and achieves state-of-the-art results on ScienceQA when ensembled with GPT-4. A key ablation showed that a two-stage training schedule—first aligning the adapter, then jointly fine-tuning the decoder—improves accuracy by 5.11%. LLaVA-1.5 [3] later replaced the linear adapter with a two-layer MLP and reached 80.34% on VQAv2.

Other VLMs. BLIP-2 [4] and InstructBLIP [5] use a Querying Transformer (Q-Former) to compress visual features before passing them to a frozen LLM, achieving strong performance (65%–79% on VQAv2). Flamingo [6] scales to 80B parameters using gated cross-attention layers interleaved throughout the LLM and achieves 82% VQAv2 accuracy. More recent models such as Idefics2-8B [7], PaliGemma [8], and Molmo2-8B push accuracy to 81–87%.

Some models have also explored efficiency techniques. MoE-LLaVA [9] (1.6B active parameters) achieves 77.6% VQAv2, showing that sparsely-activated MoE architectures can improve the accuracy-compute tradeoff. Our work takes a different approach: rather than architectural sparsity, we investigate how a purely dense, small decoder can perform when paired with a frozen CLIP encoder.

Scaling laws. Scaling laws for language models [10, 11] show that performance predictably improves as a power law with both model size and data. Our work extends this analysis to the small-model regime for VLMs, investigating four axes of scale.

4 Approach

Inspired by LLaVA [2], we start from a minimal architecture and investigate scaling along four axes to determine the best operating point. All model architecture and training code were written by the author, with the exception of scripts for visualizing plots and Modal orchestration which were generated with LLM assistance (Claude Sonnet 4.6).

4.1 Model Architecture

The model architecture (Figure 1) consists of three components:

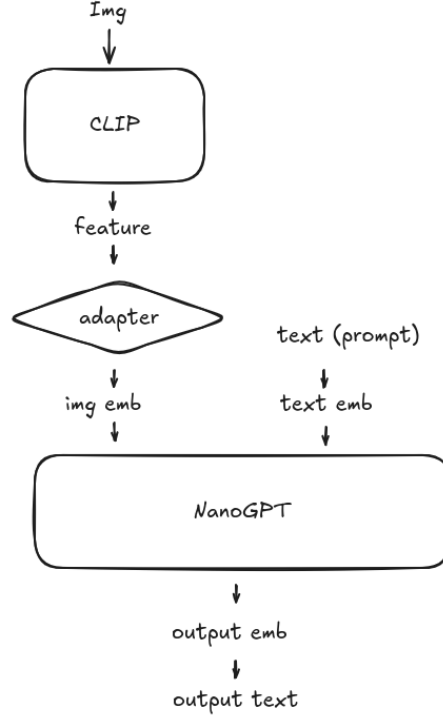


Figure 1: **NanoVQA model architecture.** A frozen CLIP encoder outputs spatial patch tokens that are projected into GPT-2’s embedding space by an MLP adapter, then prepended to the text token sequence for autoregressive decoding.

- **Vision Encoder:** To preserve spatial information, we extracted features from the penultimate layer of CLIP `vit-large-patch14`, resulting in a grid of 256×1024 tokens. This provides significantly higher fidelity than the final 1×512 class token.
- **MLP Adapter:** A small MLP projects the 1024-dimensional CLIP features into the 768-dimensional embedding space of NanoGPT.
- **Language Decoder:** We use GPT-2 (124M parameters) as the causal decoder. During training, the image embeddings are prepended to the text embeddings.

4.2 Data Pre-processing and Training

For a given image embedding X_v , question X_q , and answer X_a , the input sequence is structured as: $[X_v X_q \langle \text{ANSWER_SEP} \rangle X_a \langle \text{EOS} \rangle]$ where $\langle \text{ANSWER_SEP} \rangle$ is “`\nAnswer:`”, to delimit the shift from reading to generation. The total sequence length is 1024 tokens (256 image + 768 text).

The loss is calculated using Cross-Entropy Loss only on the X_a and $\langle \text{EOS} \rangle$ tokens. Image and question positions are masked with `ignore_index = -1` to ensure the model learns to generate the answer conditioned on visual and textual context, rather than reconstructing the input. Training follows the two-stage approach from LLaVA:

1. **Feature alignment:** Train only the adapter ($\theta = W$) with both CLIP and GPT-2 frozen. This warms up the projection before language fine-tuning.
2. **Visual instruction tuning:** Train both adapter and decoder ($\theta = \{W, \phi\}$) with CLIP frozen.

4.3 Baseline

We use NanoGPT on VQAv2 *without* any image input X_v as a text-only baseline. In this configuration, the model is tasked with predicting the answer based solely on the text of the question, providing a measure of the dataset’s language bias. All other hyperparameters are kept constant.

5 Experiments

We conduct two sets of experiments: (1) best training run on full VQAv2 dataset, and (2) scaling study on a subset to determine how performance varies with model and data choices.

5.1 Data

The VQAv2 dataset [1] contains over 265K images and 1.1M open-ended questions, each annotated with 10 human answers. The final evaluation metric is soft accuracy, $\text{Acc} = \min\left(\frac{\#\text{matching human answers}}{3}, 1\right)$. Breakdown of dataset splits is shown in Table 4.

5.2 Evaluation Method

For scaling experiments we use exact-match accuracy against the majority answer. For the final EvalAI submission we use the official VQAv2 soft-accuracy metric. Results are reported separately for four answer categories: yes/no, number, other, and overall.

5.3 Experimental Details

All experiments are run on Modal using a single NVIDIA A10G GPU (24 GiB). For the scaling study, CLIP features are pre-computed and saved; for the longer full-dataset run, CLIP features are computed on-the-fly to avoid storing cached embeddings. Training uses AdamW with betas as PyTorch defaults (0.9, 0.999). Key hyperparameters are shown in Table 5.

For scaling experiments, we vary one axis while holding the others fixed at a shared baseline of GPT-2 / MLP-64 adapter / 256 CLIP tokens / 17k small-split data:

- **Decoder size:** GPT-2 (124M), GPT-2-Medium (355M), GPT-2-Large (774M)
- **Adapter bottleneck:** Linear (no hidden layer), MLP-64, MLP-256
- **CLIP spatial tokens:** 16, 64, 256
- **Training set size:** 1K, 5K, $\sim$ 17K samples

5.4 Results

Main results. Quantitative results on the VQAv2-small split across training configurations are summarized in Table 1. The two-stage training procedure yields the best accuracy (34.43%) on the small split. Scaling to the full VQAv2 training set (443,757 samples) with a single epoch raises validation accuracy to 48.89%.

Configuration	Trainable Params	Train Loss	Val Loss	Val Accuracy
Baseline (Text-only)	124.4M	1.03	1.86	24.33%
Adapter-only	0.79M	2.88	2.58	21.11%
Adapter + Fine-tune	125.2M	0.81	1.76	34.43%
Adapter + Fine-tune (Full VQAv2)	124.6M	1.26	1.15	48.89%

Table 1: Comparison of training configurations on VQAv2-small (rows 1–3) and the full VQAv2 training set (row 4). The *Full VQAv2* model is trained for one epoch on full dataset train split.

Test-set evaluation. The full model was submitted to the VQAv2 EvalAI server for official evaluation. Results are shown in Table 2. NanoVQA achieves **60.09%** on the test-standard split, outperforming the much larger Idefics-9B (50.9%) while using over $10^4 \times$ fewer training FLOPs. Total training time on a single A10G was 5 hours 2 minutes.

We expected NanoVQA to perform around 50-60% accuracy. The current SOTA model on Molmo2-8B with 87.07% accuracy. The original LLaVA-1.5-13B model achieved around 80.34% accuracy. The VQAv2 benchmark is specifically designed to prevent language priors by memorizing yes or no answers, by using a balanced dataset and complementary image pairs where the same question yields different answers, forcing it to use visual understanding. Hence, 50-60% on an open-ended task would be a fairly significant score.

Split	Yes/No	Number	Other	Overall
test-dev	74.82%	41.23%	51.60%	59.99%
test-standard	74.81%	40.97%	51.67%	60.09%

Table 2: NanoVQA accuracy breakdown on the VQAv2 EvalAI test splits. The model performs strongest on binary yes/no questions and weakest on numerical questions, consistent with VQAv2 baseline trends.

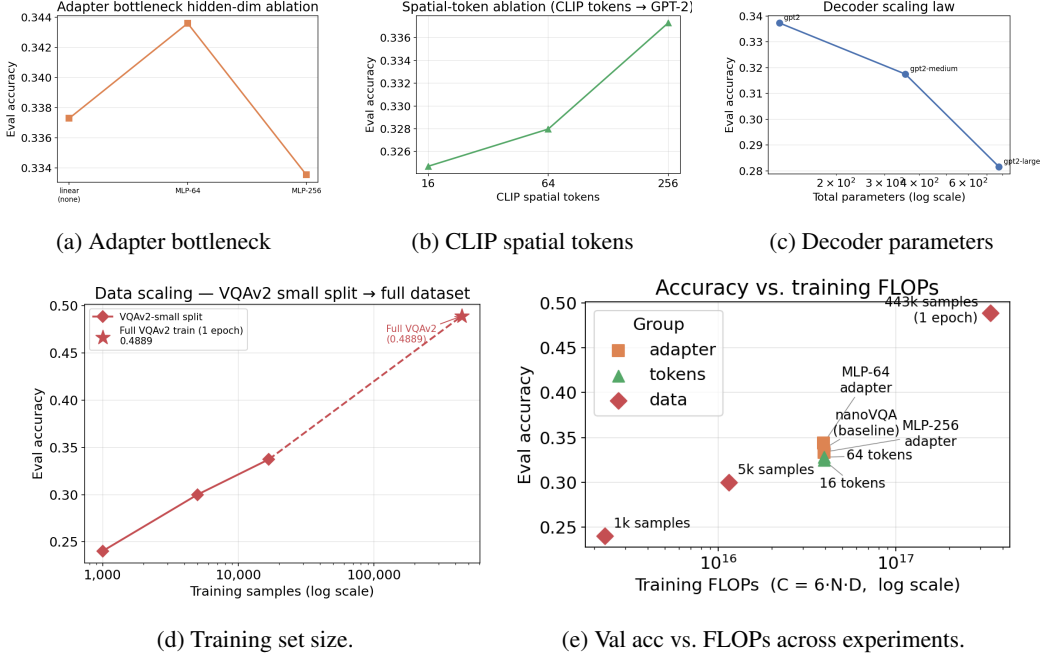


Figure 2: **Scaling analysis for NanoVQA.** Ablations over architectural and data hyperparameters. Data scaling (d) shows the strongest effect. While we attempted scaling the decoder size, this didn't yield conclusive results, as LR sweeps were needed and available GPU memory was a bottleneck to scaling up parameter size.

Scaling experiments. Results from scaling each axis are shown in Figure 2. Key findings:

- **Data scaling:** Accuracy rises from 24.03% at 1K samples to 30.00% at 5K, 33.73% at 17K, and 48.89% at 443K, following a clear log-linear trend.
- **Adapter size:** Results differ by less than a percentage. This may be due to the small hidden layer sizes used (compared to 1024 or 2048), which reduces instead of increases parameters.
- **CLIP tokens:** There is monotonic improvement from 32.47% (16 tokens) to 32.80% (64) to 33.73% (256), indicating that spatial visual detail matters which can be taken advantage of even with a small decoder.
- **Decoder size:** GPT-2-Medium (31.75%) and GPT-2-Large (28.15%) both underperform GPT-2 (33.73%). We attribute this to using the same learning rate and training only 3 epochs. Larger pretrained models likely require a lower LR and more data.

Pareto analysis. Figure 3 plots NanoVQA alongside main published models on the VQAv2 leaderboard. NanoVQA uses $\approx 3.40 \times 10^{17}$ training FLOPs and achieves 60.09% accuracy. NanoVQA lies on the lower-left Pareto frontier, achieving a favorable accuracy-compute tradeoff.

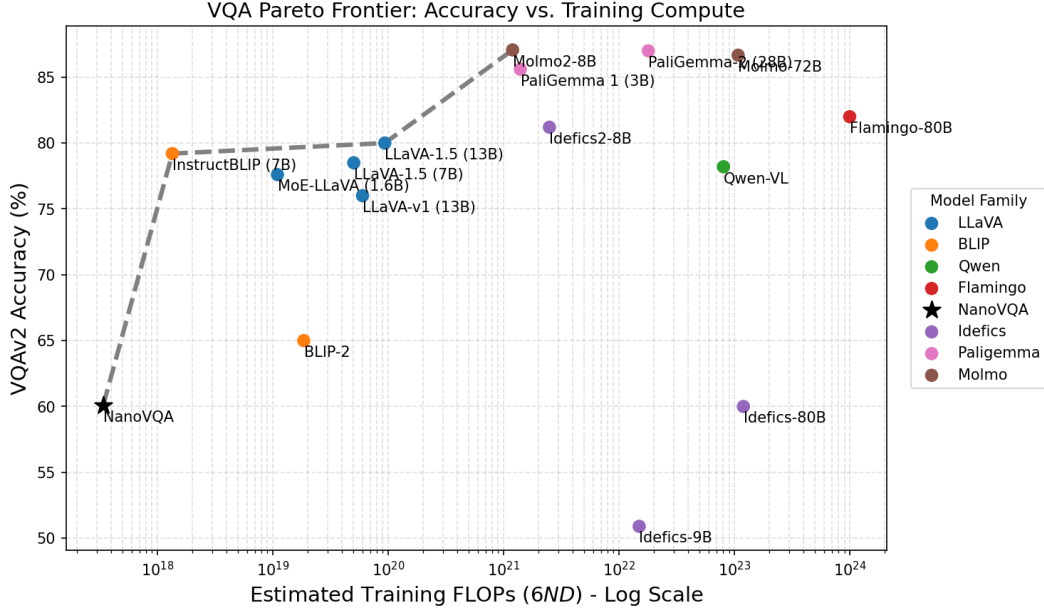


Figure 3: Pareto curve of VQAv2 accuracy vs. training compute for published VLMs. NanoVQA (black star) achieves 60.09% with 3.40×10^{17} FLOPs, 3–4 OOMs below comparable models. Training FLOPs calculated as $6ND$ where $D = \text{samples} \times \text{seq_len} \times \text{epochs}$.

6 Analysis

We analyze NanoVQA’s behavior through qualitative analysis of model outputs, error breakdown by answer category, and ablations over training decisions.

6.1 Qualitative Analysis

Table 3 shows model outputs on five randomly selected validation samples. NanoVQA correctly answers sport and yes/no questions, but still struggles with counting and spatial reasoning (also see Appendix). The text-only baseline consistently answers “no”, suggesting it may have learned biases without access to visual information.

Question	Ground Truth	NanoVQA (Ours)	Baseline (Text-Only)
Where are the kids riding?	carnival ride	street	in air
Is this boy a good pitcher?	yes	yes	no
What is the person wearing?	wetsuit	shirt	no
How many sinks in bathroom?	4	1	1
What sport are the girls playing?	soccer	soccer	no
Accuracy	—	40.0%	0.0%

Table 3: Qualitative comparison between NanoVQA and the text-only baseline on 5 randomly selected samples. The baseline seems lacks the visual context necessary to move beyond basic priors and predicts yes/no even when not applicable (row 3).

Failure modes across answer categories. The accuracy breakdown in Table 2 reveals that yes/no questions (74.81%) are much easier than “other” open-ended questions (51.67%). Numerical questions were the most difficult (40.97%). This suggests counting and quantity estimation require fine-grained details that the 256 image tokens may not capture, or a gap in model reasoning from using a smaller decoder.

Training dynamics. In initial training, there were a few common failure modes. In the adapter-only stage, the model frequently predicted `<EOS>` immediately, likely because the most statistically likely prediction to minimize loss on short VQA answers (1-2 words) is the EOS token. In the full fine-tuning stage, the model began generating content but often repeated answers in a loop (e.g., “yellow `<EOS>` yellow `<EOS>` ...”). This suggests a need for better sequence delimiters and potentially a penalty for the EOS token during the early stages of training. These were resolved by adding “\nAnswer:”, providing a clear boundary between the reading and generation phases.

6.2 Ablations

Two-stage vs. single-stage training. Figure 4 compares single-stage fine-tuning (34.12%) vs the two-stage adapter learning then jointly fine-tune schedule (34.92%). The adapter-only result shows that the MLP projection learns a coarse vision-language alignment even with a frozen decoder, exceeding the text-only baseline. Two-stage training outperforms single-stage by 0.80 percentage points, which is consistent with LLaVA’s findings but on a much smaller scale; they observed ≈ 5 percentage points difference. The two-stage advantage is modest here because our dataset size (17K samples vs. LLaVA’s 80K+), leaving less room for alignment pre-training to benefit.

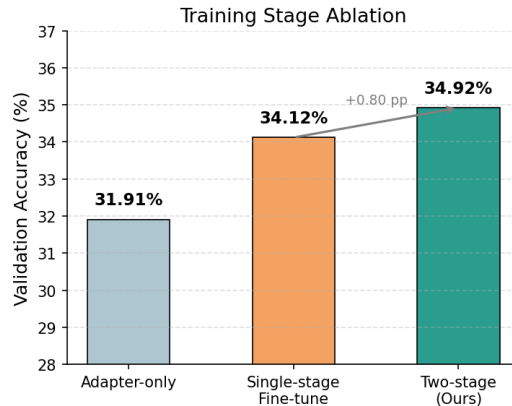


Figure 4: Training stage ablation: two-stage training (adapter pre-training then joint fine-tuning) outperforms both adapter-only and single-stage fine-tuning.

7 Conclusion

We presented NanoVQA, a minimal visual instruction-tuning framework pairing a frozen CLIP encoder with a GPT-2 decoder and a small MLP adapter. Trained on the full VQAv2 dataset for a single epoch on one A10G GPU, NanoVQA achieves 60.09% test-standard accuracy—outperforming Idefics-9B while using 4–5 orders of magnitude fewer training FLOPs—and sits on the Pareto frontier of accuracy vs. training compute.

Our scaling analysis reveals that data scale is the dominant factor in this regime: tripling training data from 17K to 443K samples raises accuracy by 15 percentage points. Adapter architecture and CLIP token count provide smaller but consistent gains, while decoder scaling may require careful LR tuning to be beneficial.

Limitations. The current adapter is a simple MLP, whereas architectures like Flamingo and BLIP-2’s Q-Former use gated cross-attention, which may allow more flexible routing of visual information. Decoder scaling experiments are inconclusive without per-model LR sweeps. Numerical question accuracy (40.97%) remains a clear weakness, likely requiring finer-grained spatial reasoning beyond what 256 patch tokens and a 124M-parameter decoder can provide.

Future work. Several extensions are promising. First, sweeping learning rates for each decoder size would yield cleaner decoder scaling curves. Second, using larger hidden layer sizes for MLP scaling would reveal the effect of less compression in the bottleneck adapter, as well as experiments

replacing the MLP adapter with cross-attention (as in Flamingo) or a Q-Former (as in BLIP-2) which may improve visual grounding at similar parameter counts. Third, increasing CLIP spatial resolution (e.g., using higher-resolution patch grids) may help with counting and localization. Furthermore, investigating how much of NanoVQA’s performance derives from CLIP’s pretrained visual representations, by comparing against a randomly initialized vision encoder, would help determine which component is doing the heavy lifting. Exploring how other techniques such as VQ-Gan or recent multi-modal models perform in VQA, and methods to improve model efficiency and reasoning, would be an interesting direction in future work.

References

- [1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering challenge. <https://visualqa.org/challenge.html>, 2026. Accessed: 2026-02-10.
- [2] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.
- [3] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024.
- [4] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.
- [5] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *NeurIPS*, 2023.
- [6] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.
- [7] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? In *NeurIPS*, 2024.
- [8] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Matthias Minderer, Maxime Oquab, Xiaohua Zhai, Olivier Hénaff, et al. PaliGemma: A versatile 3B VLM for transfer. In *CVPR*, 2025.
- [9] Bin Lin, Zhenyu Tang, Yang Ye, Jiayi Cui, Bin Zhu, Peng Jin, Jinfa Zhang, Junwu Jiang, Jianjun Luo, and Li Yuan. MoE-LLaVA: Mixture of experts for large vision-language models. In *NeurIPS*, 2024.
- [10] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.
- [11] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [12] Merve Noyan. vqav2-small. <https://huggingface.co/datasets/merve/vqav2-small>, 2023.

A Appendix

A.1 Dataset and training details

Additional details on dataset splits and hyperparameters are shown in Table 4 and 5.

Split	Samples
VQAv2 train	443,757
VQAv2 val	214,354
VQAv2 test-dev / test-standard	447,793
VQAv2-small train (80%)	17,148
VQAv2-small val (20%)	4,287

Table 4: Dataset splits. “VQAv2-small” is `merve/vqav2-small` ($26\times$ smaller) from HuggingFace [12], used for scaling experiments and ablations. The final model is trained on VQAv2 train split. Evaluation reported on EvalAI uses the `test-standard` split.

Hyperparameter	Adapter-only	Full fine-tune
Learning rate	3×10^{-3}	3×10^{-4}
Batch size	16	16
Epochs	1	3 (small) / 1 (full)
Sequence length	1024	1024

Table 5: Training hyperparameters for two-stage training.

A.2 Examples of failure modes in model sampling outputs

Qualitative analysis reveals primary failure modes in the NanoVQA sampling outputs:

1. **Spatial reasoning and layout grounding.** While the model correctly identifies object presence (e.g., surfboards, dogs), it often fails to capture relative positions or interactions (see Figure 5a–c). This suggests a bottleneck in the vision-language adapter where fine-grained spatial coordinates are lost during feature compression.
2. **Functional and logical reasoning.** The model struggles with low-level visual tasks such as precise counting, size estimation, and OCR (Figure 6a–c). These errors indicate that while the CLIP backbone provides semantic features, the decoder lacks the depth to perform complex symbolic or arithmetic operations.
3. **Perceptual confusion from ambiguous prompts.** The model may identify a correct attribute (e.g., a color) but fail to map it to the specific "surrounding" context requested (Figure 5d). This likely stems from architectural limitations in the small-scale decoder, which may struggle to parse the syntactic nuances of complex English prompts.
4. **Dataset ambiguity and label noise.** Many "failures" are actually semantically valid synonyms or grammatical variations that differ from the ground-truth annotation (Figure 7). This is an inherent limitation of the VQAv2 dataset’s evaluation metrics rather than a model deficiency and could be mitigated through more robust dataset cleaning or soft-matching.

Examples from sampling from NanoVQA responses from the validation dataset are shown below.

Image	Question	Ground Truth	Prediction	Correct?	Meta
	What is in the foreground?	rocks	surfboard	✗	idx=20085
(a) Spatial grounding (Surfboard)					
	What is the dog standing on?	water	surfboard	✗	idx=18783
(b) Interaction logic (Dog/Surfboard)					
	What's under the cap?	hair	bat	✗	idx=18149
(c) Relative positioning (Cap/Bat)					
	What color is surrounding the sink?	black	white	✗	idx=17717
(d) Contextual color identification					

Figure 5: **Failures in Spatial and Contextual Reasoning.** (a-c) model faces difficulty with layout grounding and object interactions. (d) Shows confusion between global and local surrounding colors.

	How tall is the giraffe?	very	2 feet	✗	idx=17732
(a) Size estimation					
	How many zebra?	3	2	✗	idx=20060
(b) Counting					
	What does the red sign say?	dur	stop	✗	idx=20757
(c) OCR/Reading					

Figure 6: **Functional Failures.** (a-c) Challenges in low-level vision tasks like counting, size estimation, and text recognition.

	What are they about to do?	ski	skiing	✗	idx=19829
---	----------------------------	-----	--------	---	-----------

Figure 7: **Dataset Ambiguity.** A failure case where the model’s answer is semantically correct but rejected by the dataset’s exact-match constraints.

Image	Question	Ground Truth	Prediction	Correct?	Meta
	What is the object in this photo?	(none provided)	mouse		custom image: assets/robot_arm.jpg

Image	Question	Ground Truth	Prediction	Correct?	Meta
	What is in the center of this photo?	(none provided)	clock tower		assets/ church.jpg

Figure 8: **Qualitative Zero-Shot Results.** NanoVQA failures on out-of-distribution, user-uploaded images.