Molmo and PixMo

Open Weights and Open Data for State-of-the-Art Vision-Language Models
(CVPR) 2025

Allen Institute for AI & University of Washington
Presented by: - Vedaang Chopra

Georgia Tech

Molmo and PixMo: Open Weights and Open Data for State-of-the-Art Vision-Language Models

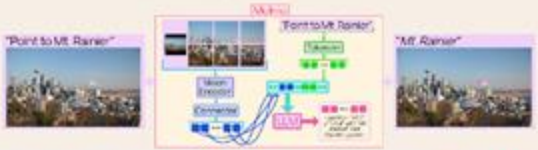
Molmo Team†

Molmo

Ai2 CVPR Nashville

Introduction

- Molmo is a SoTA VLM model trained with two-stage training on dense captioning and a mixture of PixMo and academic datasets
- PixMo is an open dataset containing millions of annotations. None of the data is distilled from proprietary VLMs to ensure an independent and transparent data pipeline
- Training Requires only 2.5k GPUs hours for a 7B model. Overlapping crops and multi-annotation image training improve efficiency and performance



PixMo

Captions

AskModelAnything

Pointing

Synthetic

Molmo

Fine-grained Understanding

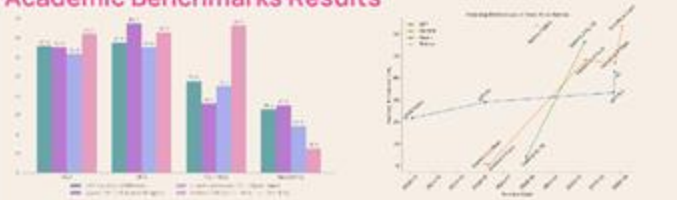
User Interaction

Pointing and Counting

Visual Skills

User Preference Results

Academic Benchmarks Results



†Built By : Matt Deitke* Christopher Clark*, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Jen Dumas, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, Aniruddha Kembhavi

Fig:- Poster Presentation of Molmo Paper

Georgia Tech

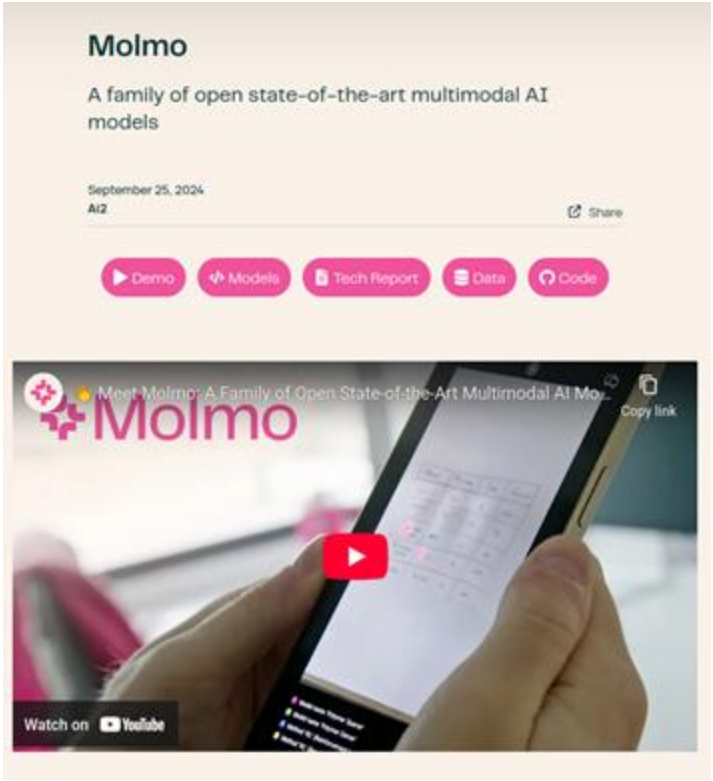
Problem Introduction and Motivation

What is the problem that this paper addresses ?

- **Problem**: - Lack of open, transparent, and high-performing vision-language models
  - Category-1: - API Based: - GPT-4o, Claude, Gemini, Groq,
  - Category-2: - Open Weights: - Qwen, InternVL, PaliGemma
  - Category-3: - Open Weights & Data: - LLava, Cambrian, Xgen

Solution:- MOLOMO

- **A state-of-the-art open VLM:** First large-scale open-weights + open-data + open-code (still the vision encoder is left out !) demonstrating competitive performance
  - **Open weights**
  - **Open data (PixMo)**
  - **Open training code**



Git Repo: - https://github.com/allenai/molmo/tree/main
Website:- https://allenai.org/blog/molmo

Georgia Tech

		VLM		LLM Backbone		Vision Encoder	
Category	Model	Open Weights	Open Data + Code	Open Weights	Open Data + Code	Open Weights	Open Data + Code
Molmo	Molmo-72B	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-D	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-O	Open	Open	Open	Open	Open	Closed
	MolmoE-1B	Open	Open	Open	Open	Open	Closed
API Models	GPT-4o	Closed	Closed	Closed	Closed	Closed	Closed
	GPT-4V	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Pro	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Flash	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3.5 Sonnet	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Opus	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Haiku	Closed	Closed	Closed	Closed	Closed	Closed
Open Weights	Qwen VL2 72B	Open	Closed	Open	Closed	Open	Closed
	Qwen VL2 7B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 LLAMA 76B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 8B	Open	Closed	Open	Closed	Open	Closed
	Pixtral 12B	Open	Closed	Open	Closed	Open	Closed
	Phi3.5-Vision 4B	Open	Closed	Open	Closed	Open	Closed
	PaliGemma 3B	Open	Closed	Open	Closed	Open	Closed
Open Weights & Data	LLAVA OneVision 72B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA OneVision 7B	Open	Distilled	Open	Closed	Open	Closed
	Cambrian-1 34B	Open	Distilled	Open	Closed	Open	Closed
	Cambrian-1 8B	Open	Distilled	Open	Closed	Open	Closed
	xGen - MM - Interleave 4B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA-1.5 13B	Open	Open	Open	Closed	Open	Closed
	LLAVA-1.5 7B	Open	Open	Open	Closed	Open	Closed

Fig: - VLM Openness Comparison. We characterize the openness of VLMs based on two attributes (open weights, open data and code) across three model components (the VLM and its two pre-trained components, the LLM backbone and the vision encoder). In addition to open vs. closed, we use the ”distilled” label to indicate that the data used to train the VLM includes images and text generated by a different, proprietary VLM, meaning that the model cannot be reproduced without a dependency on the proprietary VLM.

Georgia Tech

Let's try to understand in a way of how a model is actually built !!

Paper Flow — Understanding Molmo Like Training a Model

PixMo (Data Stage)	Molmo (Architecture Stage)	Molmo (Training Stage)	Conclusion *(What is the point of all this !!)*
• Previous Datasets(History) • Problems in previous datasets • PixMo- The new dataset • Ablations	• Background and Quick Architecture History • MOLMO architecture • Data Preprocessing • Ablations	• Pre-Training Details • Post Training Details • Ablations	• Results and Evaluation • Ablations • Conclusion • Demo • Discussion • Q&A

Georgia Tech

Stage-1: - The Data Phase

What datasets did previous architectures use ?

2020	2021	2021-22	2022
ViT (Vision Transformer)	**CLIP (OpenAI)**	**ViLT, FLAVA**	**Flamingo (DeepMind)**
Treats an image as a sequence of patches + Transformer encoder; no convolution layers. Enabled scalable visual representation learning.	Two separate encoders — one for image (ViT/ResNet) and one for text (Transformer). Trained with contrastive loss on large web data to align embeddings in a shared space.	Single Transformer that fuses patch + token embeddings early (“fusion encoder”) for cross-modal reasoning.	Introduced cross-attention “Perceiver Resampler”: a frozen vision encoder produces tokens that a large LLM attends to via learned cross-attention layers. Enables multi-image & interleaved sequences.

Georgia Tech

What datasets did previous architectures use ?

2023	2023-24	2023-24	2024
LLaVA, InstructBLIP	**Qwen-VL, InternVL**	**Qwen-VL, InternVL**	**Qwen2-VL**
Takes CLIP/BLIP-2 vision encoders + LLM; aligns them via visual instruction tuning (GPT-4-generated conversations).	End-to-end large models combining high-res vision encoders, token resampling, and multi-task heads (OCR, doc reasoning).	End-to-end large models combining high-res vision encoders, token resampling, and multi-task heads (OCR, doc reasoning).	High-performing, open-weight decoder-only LLM family (scales well, strong reasoning; drop-in backbone for VLMs).

Georgia Tech

What were the problems with Previous Models/Datasets ?

#	Problem in Previous Dataset	Problem Cause	Models or dataset affected
1	**Closed / Proprietary or Synthetic Data Loops**	Many instruction and alignment datasets were generated using GPT-4 / GPT-4V, causing “distillation of proprietary systems” and preventing reproducibility	ShareGPT4V · LLaVA · InstructBLIP · Qwen-VL-Chat · Gemini · PaLI-Gemma
2	**Noisy and Shallow Web Captions (Lack of Fine Detail)**	Web-scraped alt-text is short (~5–10 words), inconsistent, and object-level grounding is poor. This weakens fine-grained reasoning and counting.	CLIP · ALIGN · LAION-400M/5B · DataComp · OpenCLIP
3	**Limited Grounding and Spatial Reasoning Data**	Earlier grounding/counting sets are small, single-target, or too easy, while web-scale data lacks coordinates	CLIP · ALIGN · RefCOCO · RefCOCOg · CountBenchQA
4	**Costly and Low-Quality Human Captions**	Human annotation expensive; workers produce short, repetitive, copy-pasted captions (~11 words avg)	COCO · Visual Genome · Flickr30k · CC3M/CC12M
5	**Missing Non-Photographic Modalities (Docs / Charts / Clocks)**	Prior datasets mostly natural photos; lack structured visual reasoning (documents, charts, diagrams, time)	CLIP · LAION · BLIP-2 · FLAVA · ViLT
6	**Lack of Truly Open and Reproducible Pipelines**	Large VLMs trained on closed or undisclosed data; unclear preprocessing → no reproducibility	Flamingo · ALIGN · Gemini · GPT-4V · PaLI-Gemma
7	**Insufficient Multimodal Breadth and Balance**	Strong English and photographic bias; weak multilingual and multi-domain coverage	LAION · ALIGN · DataComp

Part A:- Data Collection

PixMo (Pixels for Molmo)

01	**PixMo-Cap** Dense captions for pre-training	• **Goal**: Teach broad visual understanding with very detailed descriptions.
02	**PixMo-Ask Model Anything** Free-form Q&A for instruction following	• **Goal**: Teach the model to answer diverse, realistic questions grounded in the image.
03	**PixMo- Points** Pointing for grounding & counting	• **Goal-1**: - *Grounding:* point to items described by text • **Goal-2**: - *Counting:* count by pointing each instance • **Goal -3**: - *Explanations:* use points as visual evidence in answers
04	**PixMo- CapQA** QA from dense captions	• **Goal**: Expand QA coverage cheaply without VLMs.
05	**PixMo- Docs** Code-generated docs/charts/tables/diagrams	• **Goal**: Teach OCR, chart/table reasoning, and doc understanding.
06	**PixMo- Clocks** Realistic time reading	• **Goal**: Robust time-telling from analog watch faces.
07	**PixMo- Count** Open-domain counting with point supervision	• **Goal:** Counting across diverse web images.

Georgia Tech

PixMo-CAP

Goal: Teach broad visual understanding with very detailed descriptions.

How it's built:

- Images sourced across ~70 topics (street signs, memes, food, drawings, websites, blurry photos, …).
- Annotators speak descriptions for 60–90s (voice forces more detail and prevents copying from VLMs).
- Audio → ASR transcripts → a text-only LLM cleans/summarizes to a final caption (remove fillers, unify style).

Scale & stats:

- 712k images, 1.3M transcripts/captions; ~196 words/caption (vs 11 in COCO; 37 in Localized Narratives).

Why it's novel/useful: The voice-first trick yields richer, denser content and auditability (audio receipts), crucial for learning fine detail.

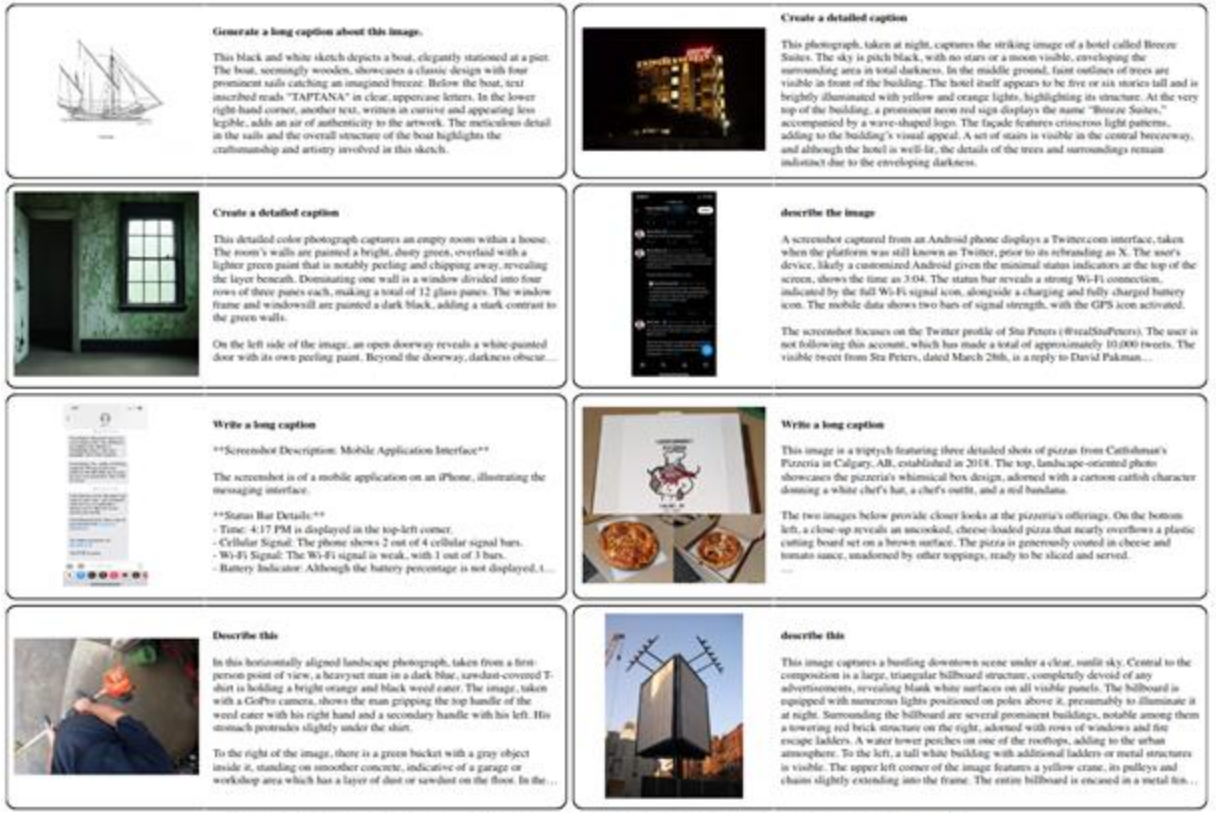


Figure 12. Randomly selected examples from **PixMo-Cap** with our prompt templates.

PixMo-AskModel Anything

Goal: Teach the model to answer diverse, realistic questions grounded in the image.

How it's built:

- Annotator picks an image and writes a question.
- Run OCR (non-VLM) + a PixMo-Cap-trained captioner.
- A text-only LLM drafts an answer using only OCR + caption (no VLM supervision).
- Annotator accepts/rejects/revises until correct.

Scale: 162k QA pairs over 73k images.

Why it matters: Human-in-the-loop yields high-quality, grounded answers without VLM dependency.

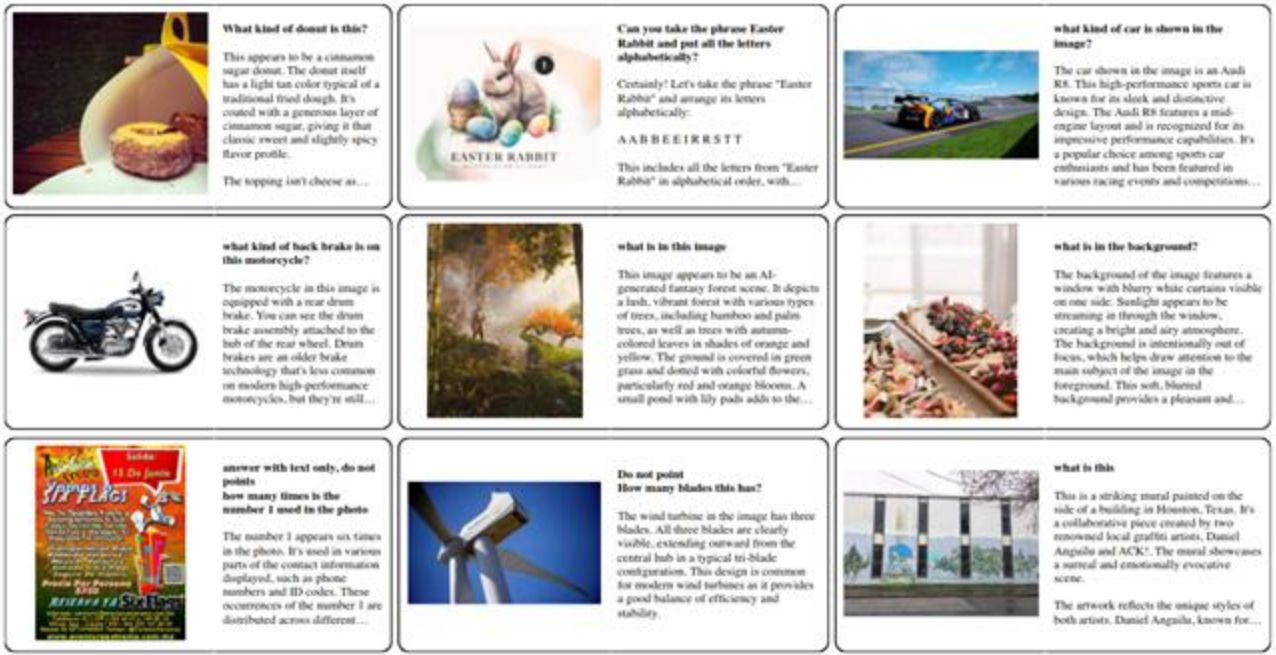


Figure 13. Randomly selected examples from **PixMo-AskModelAnything**.

Georgia Tech

PixMo-Points

Goal: To teach Molmo how to ground text in visual evidence, count objects, and explain answers visually by pointing to the exact regions in an image

How it is built: - Annotators write a short referring phrase → point to each instance → mark "not-present" if absent. Extended pipeline adds text-annotated points so LLM uses them in explanations.

Scale & stats:

- Core pointing: 2.3M question–points over 223k images (main text)
- Data detail section: 229k images, 1.98M referring expressions, 8.7 expressions/image, 5.5 points/expression, ~47.7 points/image, 359k "no-target" instances.
- 79k point-explanation annotations on 14k images.

Why it's novel/useful: ≈ 10 × larger than RefCOCO/gRefCOCO; points = faster than boxes / masks; enables "count-by-pointing" chain-of-thought and visual explainability.

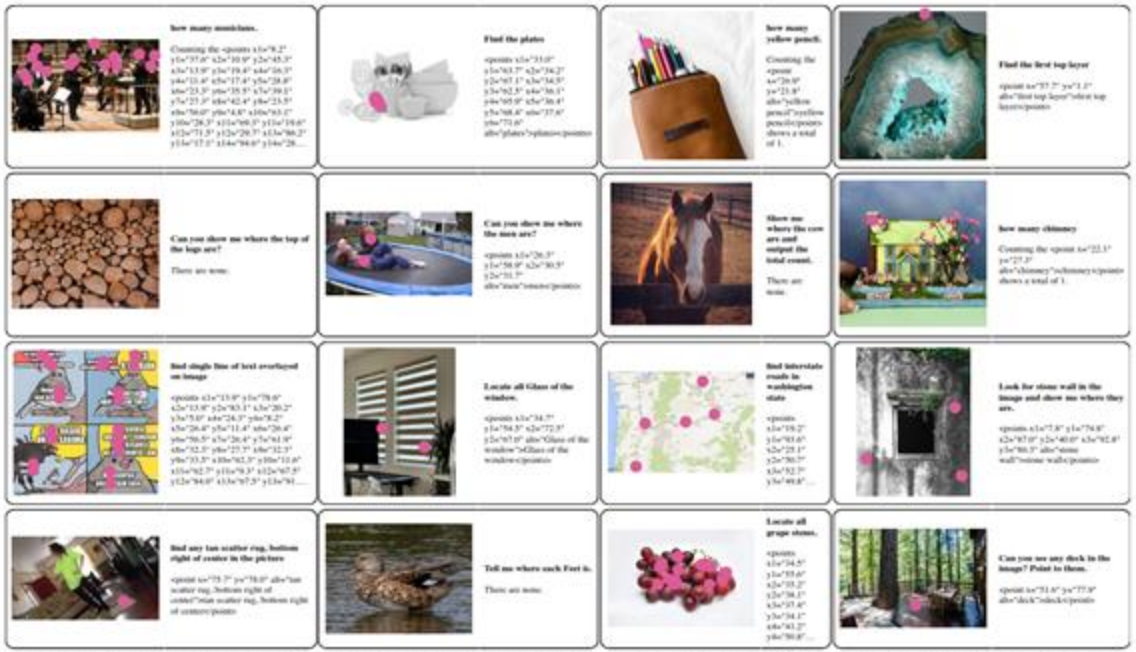
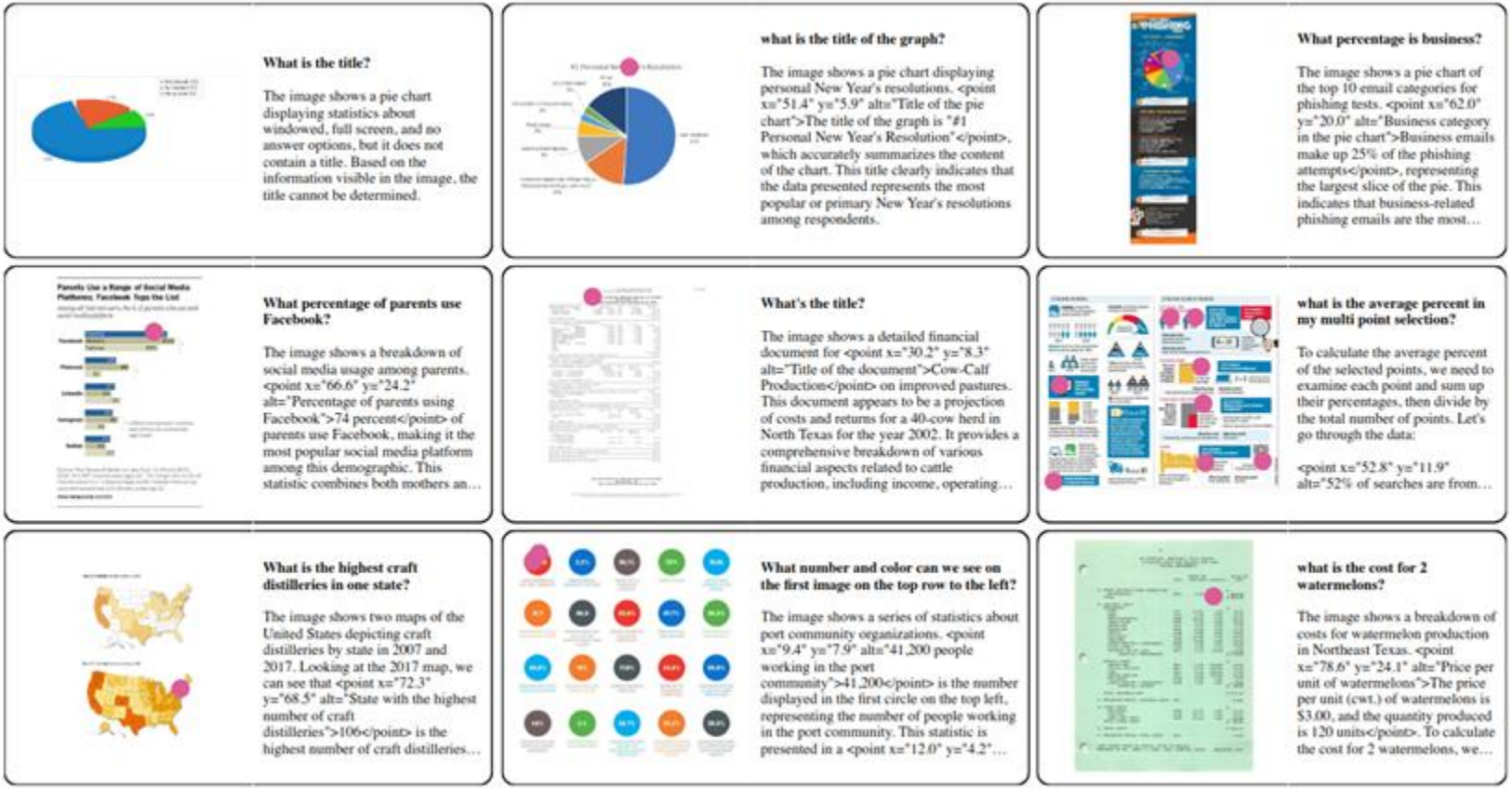


Figure 14. Randomly selected examples from **PixMo-Points**. Even when text has been cut off, all points still appear in the image. Our templated prompts can be ungrammatical for some of these options, but we find they are still sufficient to let the model respond correctly to natural language instructions.

PixMo-Points



What is the title?

The image shows a pie chart displaying statistics about windowed, full screen, and no answer options, but it does not contain a title. Based on the information visible in the image, the title cannot be determined.

what is the title of the graph?

The image shows a pie chart displaying personal New Year's resolutions. <point x="51.4" y="5.9" alt="Title of the pie chart">The title of the graph is "#1 Personal New Year's Resolution"</point>, which accurately summarizes the content of the chart. This title clearly indicates that the data presented represents the most popular or primary New Year's resolutions among respondents.

What percentage is business?

The image shows a pie chart of the top 10 email categories for phishing tests. <point x="62.0" y="20.0" alt="Business category in the pie chart">Business emails make up 25% of the phishing attempts</point>, representing the largest slice of the pie. This indicates that business-related phishing emails are the most…

What percentage of parents use Facebook?

The image shows a breakdown of social media usage among parents. <point x="66.6" y="24.2" alt="Percentage of parents using Facebook">74 percent</point> of parents use Facebook, making it the most popular social media platform among this demographic. This statistic combines both mothers an…

What's the title?

The image shows a detailed financial document for <point x="30.2" y="8.3" alt="Title of the document">Cow-Calf Production</point> on improved pastures. This document appears to be a projection of costs and returns for a 40-cow herd in North Texas for the year 2002. It provides a comprehensive breakdown of various financial aspects related to cattle production, including income, operating…

what is the average percent in my multi point selection?

To calculate the average percent of the selected points, we need to examine each point and sum up their percentages, then divide by the total number of points. Let's go through the data:

<point x="52.8" y="11.9" alt="52% of searches are from…

What is the highest craft distilleries in one state?

The image shows two maps of the United States depicting craft distilleries by state in 2007 and 2017. Looking at the 2017 map, we can see that <point x="72.3" y="68.5" alt="State with the highest number of craft distilleries">106</point> is the highest number of craft distilleries…

What number and color can we see on the first image on the top row to the left?

The image shows a series of statistics about port community organizations. <point x="9.4" y="7.9" alt="41,200 people working in the port community">41,200</point> is the number displayed in the first circle on the top left, representing the number of people working in the port community. This statistic is presented in a <point x="12.0" y="4.2"…

what is the cost for 2 watermelons?

The image shows a breakdown of costs for watermelon production in Northeast Texas. <point x="78.6" y="24.1" alt="Price per unit of watermelons">The price per unit (cwt.) of watermelons is $3.00, and the quantity produced is 120 units</point>. To calculate the cost for 2 watermelons, we…

Figure 15. Randomly selected examples from the experimental **PixMo-Points** data that includes points with explanations.

Georgia Tech

PixMo- CAPQA

Goal: Give Molmo large-scale question–answer data so it can perform interactive, question-answer style reasoning about images

How it's built: A text-only language model (LLM) is prompted to ask and answer its own questions using only the caption text as context.

Scale: 214k QA over 165k images.

Use: Adds natural question–answer format supervision that improves Molmo's dialog and reasoning abilities.

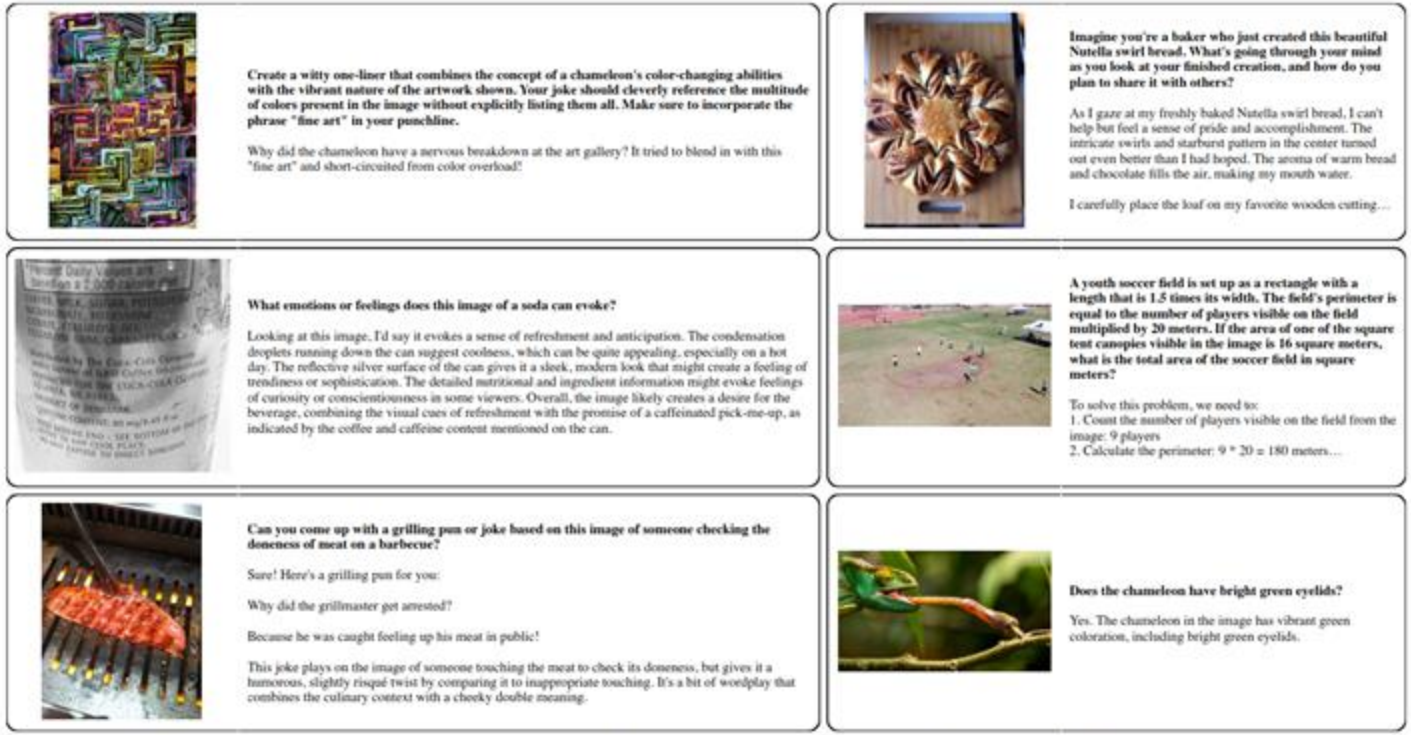


Figure 16. Randomly selected examples from the synthetic **PixMo-CapQA** data generated from PixMo-Cap.

Georgia Tech

PixMo-Docs

Goal: Teach OCR, chart/table reasoning, and doc understanding.

How it's built (two-stage, all text-LLMs, no VLMs):

An LLM writes code that renders images (charts, tables, diagrams, mixed documents). Tooling: Matplotlib, Plotly, LaTeX, HTML, Vega-Lite, Mermaid, Graphviz, Another LLM has privileged access to the code (not the image) to generate QA pairs with exact ground truth.

Scale & stats: 255k images, ~2.3M QA.

Use: - Instruction-tuning role: Provides the bulk of structured-reasoning supervision for Molmo during fine-tuning.

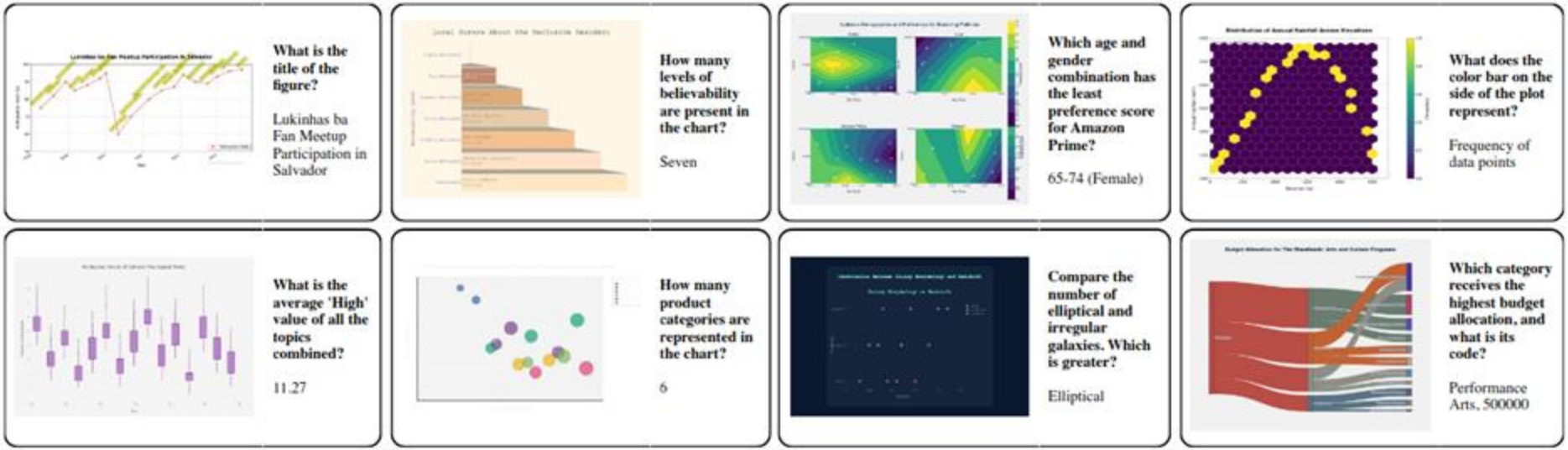


Figure 19. Randomly selected chart examples from the synthetic **PixMo-Docs** data.

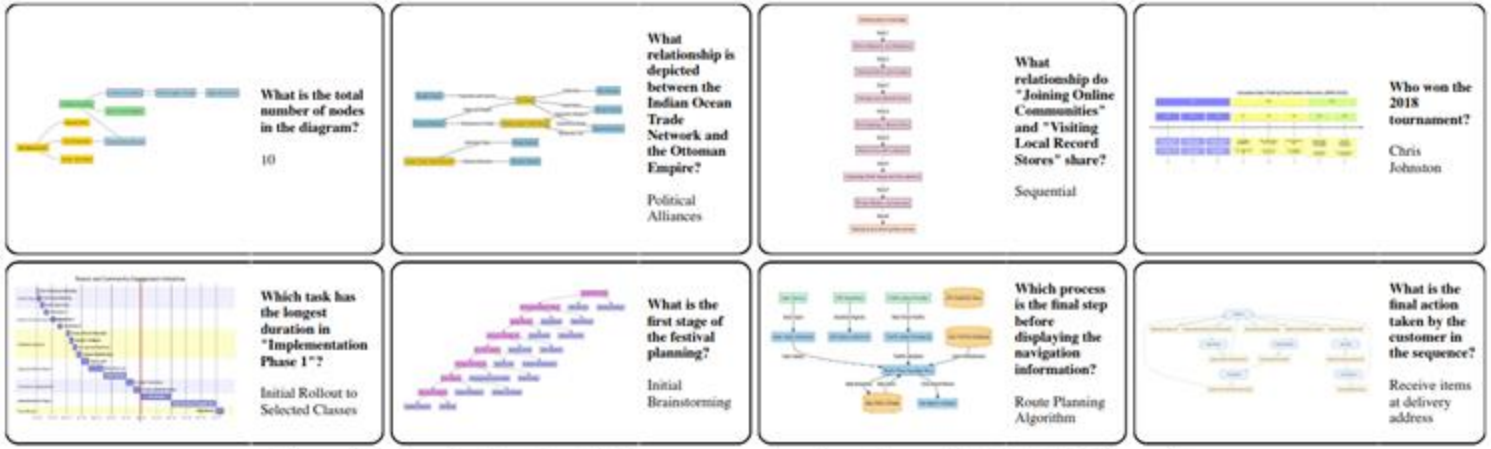


Figure 21. Randomly selected diagram examples from the synthetic **PixMo-Docs** data.

Figure 22. Other randomly selected documents from the synthetic **PixMo-Docs** data.

PixMo- Clocks

Goal: Teach Molmo to interpret analog watches → map hand positions to numerical time.

How it is built: - Programmatically render ~50 watch bodies × ~160 k faces set to random times; each image paired with QA ("What time is it?").

Scale & stats: 826 k examples (image + QA pair) · 50 body templates · 160 k faces · labels = exact HH:MM times.

Why it's novel/useful: Realistic, photo-style watches with shadows & decorations → harder than simulator datasets; links visual geometry to numerical reasoning.

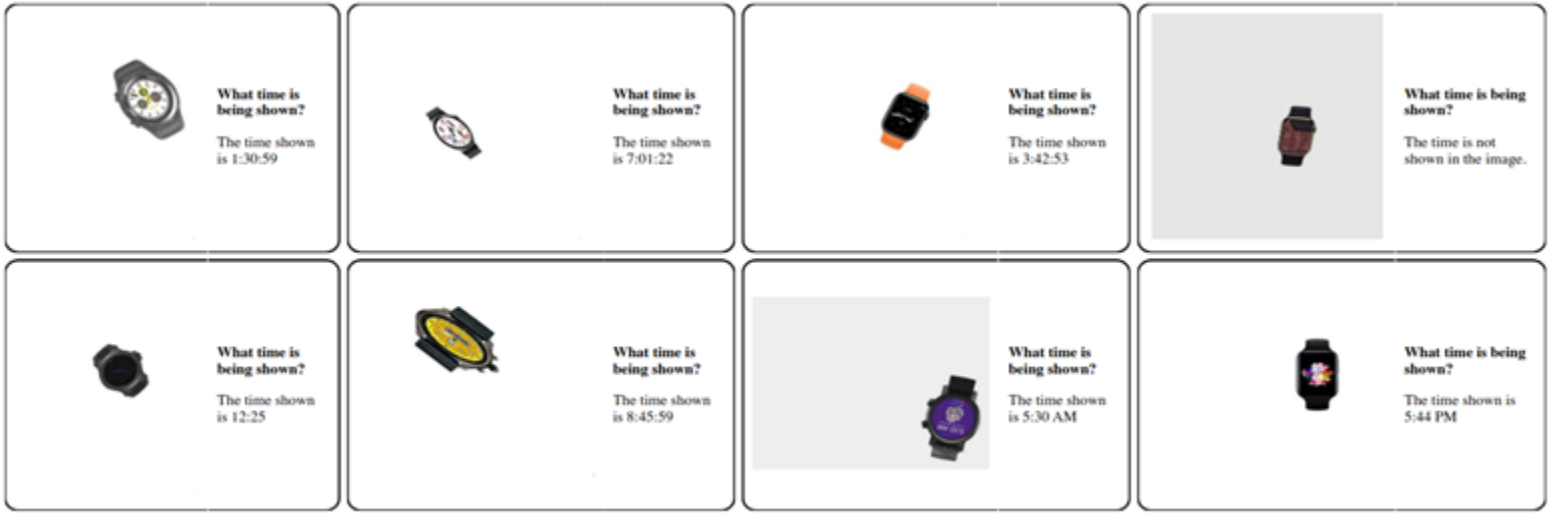


Figure 17. Randomly selected examples from the synthetic **PixMo-Clocks** data after our data augmentation.

Georgia Tech

PixMo-Count

Goal: A synthetic but realistic dataset that focuses on grounding, counting, and visual explanations via explicit 2-D pointing.

How it is built: - Diverse web images collected across many object categories and environments. Run a non-VLM, OCR model over the images to locate objects. For each image, identify the object class with the most detections (e.g., “cars” if most detections are cars). Record the count of that class (from 0–10). Use object centers as point annotations for each detected instance. Automatically form a question–answer pair such as: Q: “How many cars are in the image?” A: “5.”

Scale & stats: 36 k train images (0–10 counts) · 540 val + 540 test (verified).

Why it’s novel/useful: Adds point-level supervision for counting · harder & more diverse than CountBenchQA · enables explainable “count-by-pointing.”

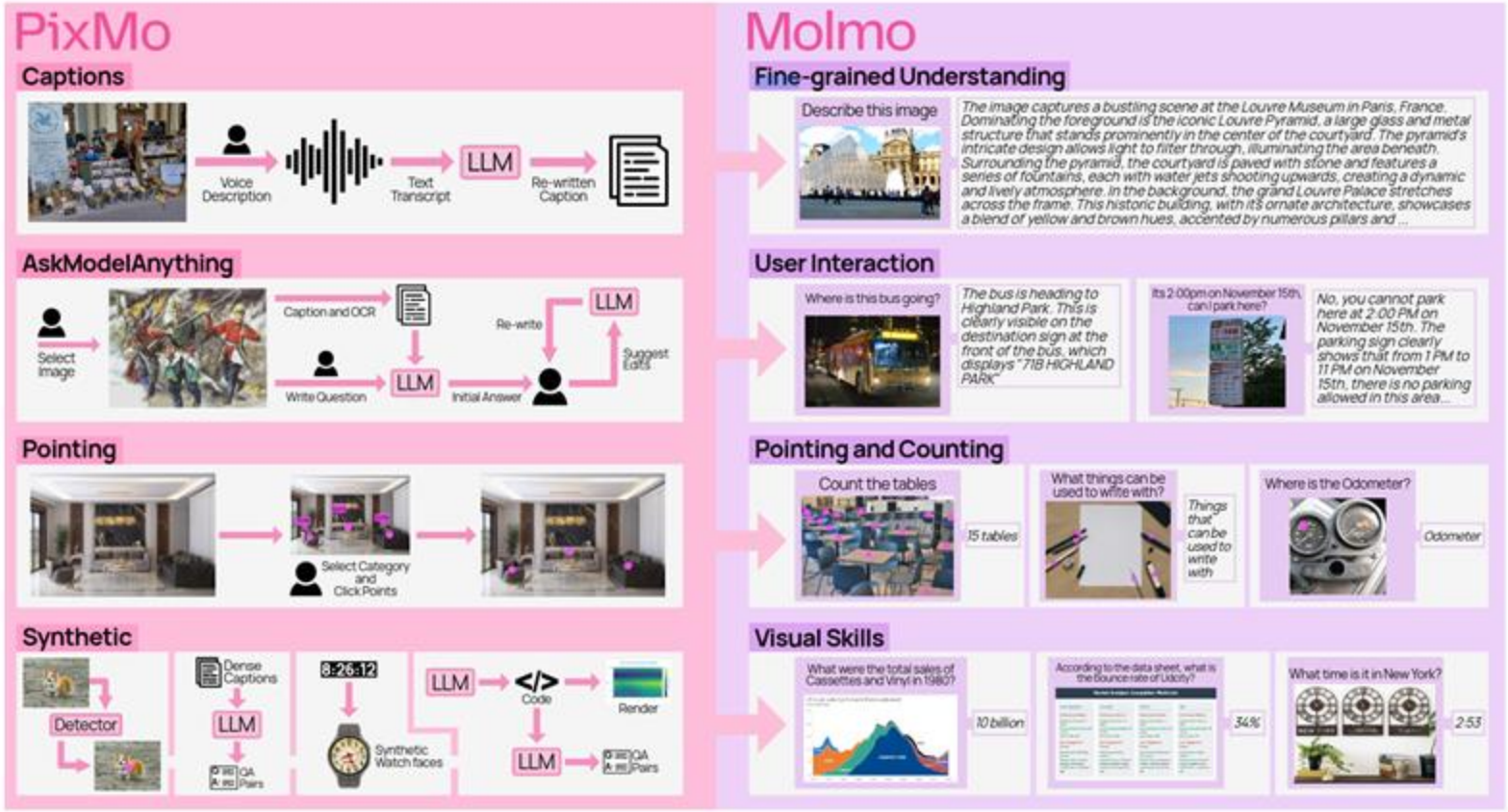
Figure 18. Randomly selected examples from the synthetic **PixMo-Count** data.

Georgia Tech

What were the problems with Previous Models/Datasets ?

#	Problem in Previous Dataset	How does PixMo Solve it ?
1	**Closed / Proprietary or Synthetic Data Loops**	All PixMo data gathered **without using any vision models** — from human speech, language-only LLM summarization, or direct programmatic generation
2	**Noisy and Shallow Web Captions (Lack of Fine Detail)**	**PixMo-Cap** uses spoken human descriptions (~200 words avg) → richer, contextual, fine-grained captions
3	**Limited Grounding and Spatial Reasoning Data**	**PixMo-Points** (2.3 M point annotations) + **PixMo-Count** (hard counting set) enable spatial and numerical reasoning
4	**Costly and Low-Quality Human Captions**	**Spoken-caption pipeline:** annotators narrate images for 60–90 s; transcribed + LLM-summarized → long, natural captions
5	**Missing Non-Photographic Modalities (Docs / Charts / Clocks)**	**PixMo-Docs & PixMo-Clocks** — code-rendered charts, tables, diagrams, clocks from LaTeX / Matplotlib / Plotly / Vega-Lite
6	**Lack of Truly Open and Reproducible Pipelines**	Molmo releases **full PixMo data + generation code**, ensuring end-to-end reproducibility and transparency.
7	**Insufficient Multimodal Breadth and Balance**	PixMo covers **multiple domains (images + documents + charts)** and supports multilingual captions, balancing modality and language diversity.

What does each subset of PixMo add to the model ?



Georgia Tech

Any Questions ?

Stage-2: - The Modelling Phase

Georgia Tech

Background and Related Works

How did the previous architectures look like ?

2020	2021	2021-22	2022
ViT (Vision Transformer)	**CLIP (OpenAI)**	**ViLT, FLAVA**	**Flamingo (DeepMind)**
Treats an image as a sequence of patches + Transformer encoder; no convolution layers. Enabled scalable visual representation learning.	Two separate encoders — one for image (ViT/ResNet) and one for text (Transformer). Trained with contrastive loss on large web data to align embeddings in a shared space.	Single Transformer that fuses patch + token embeddings early (“fusion encoder”) for cross-modal reasoning.	Introduced cross-attention “Perceiver Resampler”: a frozen vision encoder produces tokens that a large LLM attends to via learned cross-attention layers. Enables multi-image & interleaved sequences.

Georgia Tech

How did the previous architectures look like ?(contd..)

2023	2023-24	2023-24	2024
LLaVA, InstructBLIP	**Qwen-VL, InternVL**	**Qwen-VL, InternVL**	**Qwen2-VL**
Takes CLIP/BLIP-2 vision encoders + LLM; aligns them via visual instruction tuning (GPT-4-generated conversations).	End-to-end large models combining high-res vision encoders, token resampling, and multi-task heads (OCR, doc reasoning).	End-to-end large models combining high-res vision encoders, token resampling, and multi-task heads (OCR, doc reasoning).	High-performing, open-weight decoder-only LLM family (scales well, strong reasoning; drop-in backbone for VLMs).

Georgia Tech

Model Architecture

Molmo: The Architecture

- Molmo is a Vision-Language Model (VLM) — it takes an image + text input and produces text output (a caption, answer, explanation, or coordinates).

- It's built in four main blocks:
  a. Preprocessor – prepares the image (multi-scale cropping).
  b. Vision Encoder (ViT) – turns images into patch-level features.
  c. Connector – projects visual features into the same space as words.
  d. Language Model (LLM) – generates text from those tokens.

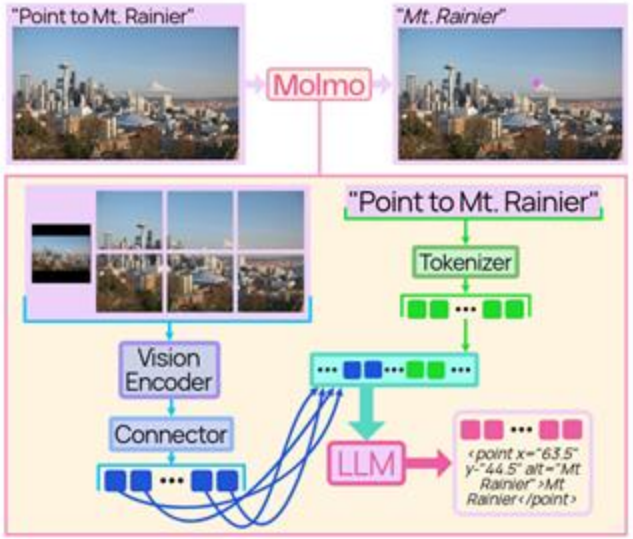


Figure 2. **Molmo** follows the simple and standard design of connecting a vision encoder and a language model.

What pre-processing on images Molmo does?

PROBLEM: -

- Vision Transformers (like CLIP's ViT-L/14) have a strict input rule: They only accept square images of a fixed resolution (for example 336 × 336 pixels).
- But real-world photos are rectangular, have different resolutions, and often contain small details (like text on signs, buttons, clocks, charts). So if we just resized everything to 336×336:
  a. Small details would blur or disappear.
  b. Wide/tall scenes would get stretched or squished.

Figure 3. An image cropped without (left) and with (right) overlap. Highlighted regions show areas used by the LLM. Overlapping crops ensure that central patches are encoded with neighboring context; for example, the patches containing the bike's brand name are always part of a crop where the entire name is visible.

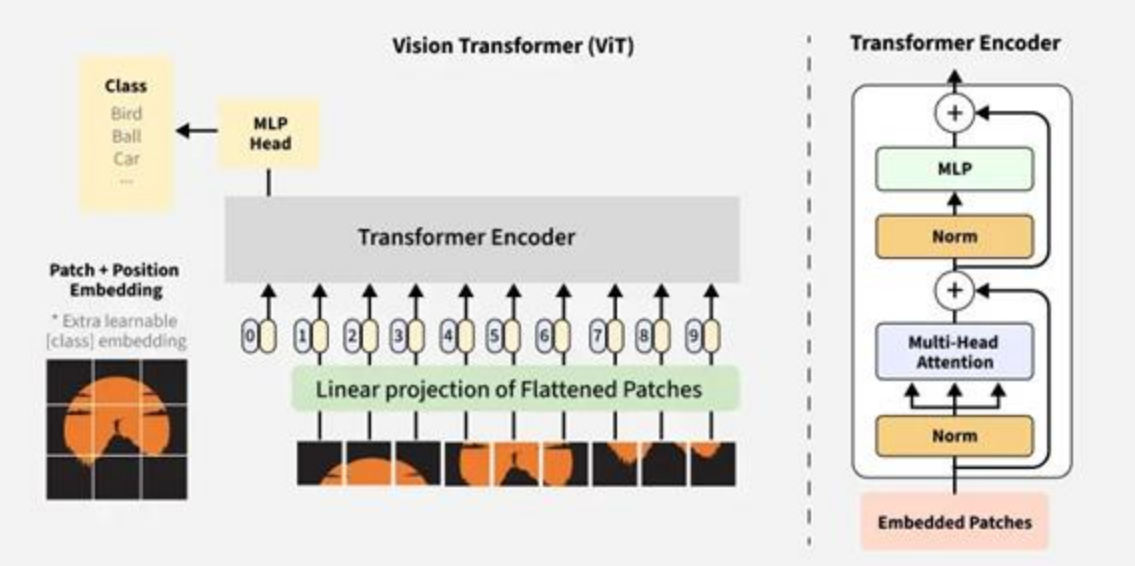
Solution: - Molmo fixes that with a multi-scale tiling strategy—the Preprocessor. We pass multiple inputs to encoder.

1. We will compress the image to low level 336*336 px for global important information
2. We will cut the image into several parts, each cut is 336*336 px, where cuts overlap each other so that information is sent to the encoder properly

Georgia Tech

Molmo: Vision Encoder

- The Vision Encoder is the part that turns raw image pixels into a set of meaningful numeric tokens that represent the image's contents — texture, shape, objects, text, and layout.
- Molmo uses a Vision Transformer (ViT-L/14, 336 px) — the same model used in CLIP — but it adds some special tweaks to make it work better for fine-grained multimodal understanding.
- Molmo Vision Encoder(variants)-
  - OpenAi; ViT-L/14 336px CLIP model
  - SigLiP
  - MetaCLIP



Molmo: Connector and LLM Decoder

- The connector bridges the ViT and the LLM.
- Takes pooled patch vectors from ViT.
- Uses a small MLP (multi-layer perceptron) to map them into the LLM's embedding space (so "visual tokens" and "word tokens" live in the same world).
- Adds positional information so the LLM knows where in the image each token came from.

- The LLM is a decoder-only transformer, like GPT-style models.
- The LLM takes input as [Vision tokens] + [Text prompt tokens]
- The LLM auto-regressively generates text, one token at a time, conditioned on both image and text context.
- LLM's, used by Molmo: -
  - OLMo-7B-1024 preview (open source)
  - OLMoE-1B-7B (most efficient from allenai)
  - Qwen2 7B (best results)

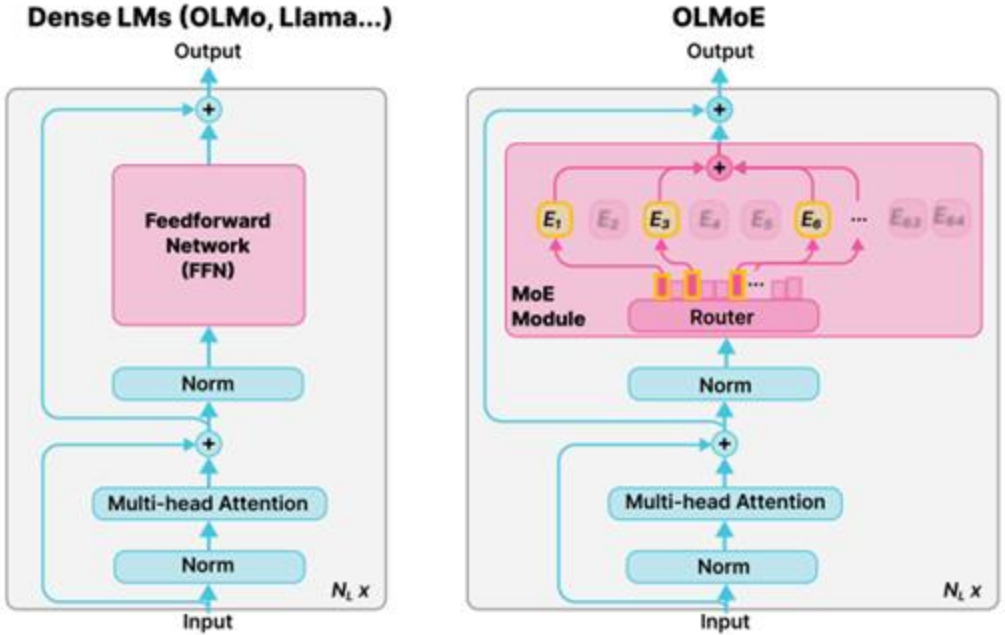


Figure 2: **Comparison of the architecture of dense LMs and MoE models like OLMoE.** The figure excludes some details, e.g., **OLMoE-1B-7B** also uses QK-Norm (§4.2.5).

Georgia Tech

How does the working look like in MOLMO ? (example)

💡 Step 1: The Input

- Real-world image: **1920 × 1080 × 3 (RGB);** *An image of a busy café street — "Café Roma" signboard, tables, people, and parked cars.*
- It has text ("Café Roma"), small details (menu board), and many objects (chairs, people).

🧠 Step 2: Making the Image ViT-Friendly

Molmo can't feed this rectangular image directly to the Vision Transformer (ViT), because ViT only works on **square 336×336** images.

So, Molmo creates:

- **1 low-resolution image** → the entire scene scaled down to 336×336 (gives global context).
- **8–12 high-resolution crops** → zoomed-in squares (336×336 each) that cover every part of the image.

Each crop overlaps its neighbor by about **56 pixels**, so borders (like "Café") don't get cut in half.

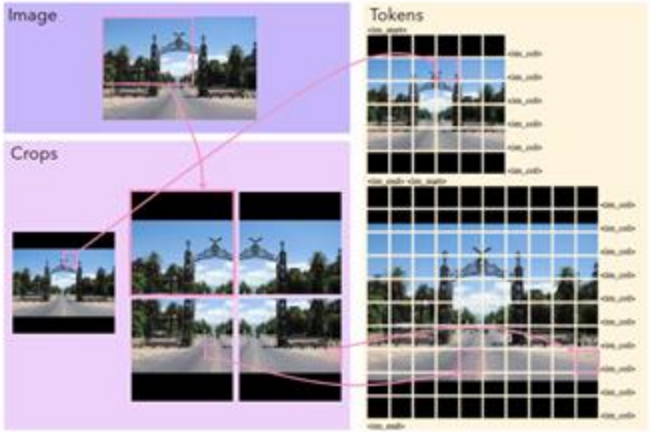


Figure 5. **Converting an image into tokens.** The image (top left) is turned into a single low-res and several overlapping high-res crops (bottom left). Padding (the black borders) is used so each crop is square and the aspect ratio of the image is preserved. The final token sequence for the image (right, arranged top-down left-to-right with line breaks for clarity) is built by extracting patch-level features from the crops, shown here using images of the patches, and special tokens. An image start and image end token are placed before/after the high-res and low-res patches, and column tokens are inserted after each row of patches. This example uses 4 high-res crops and extracts features from 36 (6×6) patches per crop, in practice Molmo typically uses 12 high-res crops and extracts features from 144 (12×12) patches per crop.

Georgia Tech

How does the inference look like in MOLMO ?

🧱 Step 3: Padding the Edges

If the grid doesn't perfectly fit, black padding is added to fill extra space.
Molmo tells the ViT whether each patch is:

- real image region,
- partially padded, or
- all padding (by adding **padding-type embeddings**).

✅ This ensures the model doesn't confuse black borders with actual dark areas of the image.

🔍 Step 4: ViT Patchification and Feature Extraction

Each crop (336×336) is divided into **14×14 px patches**, so each crop becomes a **24×24 grid = 576 patches**.

Every patch → converted to a **1024-dimensional feature vector** by ViT's patch embedding layer.

Example (per crop):

```
Input: [336, 336, 3]
↓
Split into patches → [24, 24, 1024]
↓
Flatten → [576, 1024]
```

Molmo takes ViT outputs from **two internal layers** — one mid-level (for textures), one late (for semantics) — and combines them → slightly better detail understanding.

Georgia Tech

How does the inference look like in MOLMO ?

✨ Step 5: 2×2 Attention Pooling

Now, 576 tokens per crop is too many.
So Molmo uses **2×2 attention pooling** to compress information while keeping local context.

Every 4 neighboring patches → 1 pooled token:

```
24×24 → 12×12 = 144 tokens per crop
```

Each token still has **1024 dimensions**, but now represents a *small region* (like a person's face or part of a table).

🧹 Step 6: Removing Redundant Overlaps

Since crops overlapped, some tokens describe the same pixels twice.
Molmo removes these duplicate areas, keeping only **unique patches** for the full image.

So if 9 crops × 144 = 1296 tokens before cleanup,
after removing overlap → roughly **1100 unique visual tokens** remain.

🎯 Step 7: Vision–Language Connector (The Bridge)

Each vision token is a **1024-D vector** (from ViT),
but our LLM (Qwen2 or OLMo) uses **4096-D embeddings** for text.

So Molmo adds a small **MLP connector** that maps:

```
[1100, 1024] → [1100, 4096]
```

Now all vision tokens "look" like text tokens — just numbers in the same space.

Georgia Tech

How does the inference look like in MOLMO ?

🧩 Step 8: Add Layout Tokens

To tell the LLM how the image was tiled, Molmo adds **special layout tokens**:

```
<img_start_lowres> ... <img_end_lowres>
```

```
<img_start_hires> ... <row_end> ... <img_end_hires>
```

This helps the model “know” that one token sequence came from the top-left crop, another from bottom-right, etc.

Final vision sequence length: about **1110 tokens** (4096-D each).

💬 Step 9: Add the Text Prompt

Now the user asks a question —“What color is the car parked near the café?”

These words are tokenized into ~8 text tokens (4096-D each).

Molmo **concatenates**:

```
[Vision tokens][Text tokens]
```

```
→ [1110 + 8 = 1118 tokens, 4096-D each
```

Georgia Tech

How does the inference look like in MOLMO ?

⚙️ Step 10: LLM Forward Pass (Decoder-Only Transformer)

Inside the LLM:

- Vision tokens → **context memory** (can look at each other freely).
- Text tokens → **causal** (each new word can attend to all vision tokens + previous text).

Now self-attention learns relationships like: So during generation, when predicting the next token, the model "looks back" at the vision embeddings representing those regions.

Text Token	Attends to Vision Tokens
"car"	high attention on the car region
"color"	focuses on same area
"café"	attention on signboard
"?"	weak uniform attention

📜 Step 11: Output

The decoder outputs the next tokens one by one:

```
Vision + "What color is the car?"
↓
LLM attends to car patches
↓
Predicts "red"
↓
"The car is red."
```

That's how Molmo **connects visual understanding to language reasoning.**

Georgia Tech

Molmo: Architecture(Ablations)

ViT-L/14	cap F_1	11-avg
OpenAI CLIP 336px	54.1	76.9
MetaCLIP 336px	54.1	**77.2**
SigLIP-So400m 384px	**54.4**	77.1
DINOv2 336px	53.2	75.6

(a) **Vision encoder**. Encoders that were trained on noisy web-scale image-text pairs perform similarly (rows 1-3). Surprisingly, DINOv2, which is trained on images only (no text, no label supervision), is competitive on these tasks. MetaCLIP and DINOv2 are *fully* open.

# crops train, test	cap F_1	11-avg
4, 4	52.0	71.0
4, 12	52.0	74.1
4, 36	52.0	74.2
12, 12	54.1	74.9
12, 36	**54.1**	76.9
36, 36	54.0	**77.2**

(b) **Image resolution**. Using more crops at training and testing time generally improves performance. However, captioning and counting can perform poorly when # of crops are unequal, so for these tasks we always set the number of test crops equal to the training value.

pre-train, fine-tune	cap F_1	11-avg
off, off	53.1	74.6
off, on	53.1	76.6
on, on	53.7	**77.0**
on (text only), on	**54.1**	76.9

(c) **Dropout**. Dropout in the LLM improves pre-training and fine-tuning results. In pre-training, applying dropout to captioning text tokens only further improves results. This design may encourage the model to rely more on vision tokens rather than past text tokens.

cropping	cap F_1	11-avg
single	46.7	62.8
multi, no overlap	53.4	75.7
multi, overlap	**54.1**	**76.9**

(d) **Cropping**. Using the entire image only (single crop) performs poorly. Our novel overlapping crop method (see Figure 3), which prevents loss of context, performs the best.

setting	cap F_1	11-avg
off	53.0	76.2
on	**54.1**	**76.9**

(e) **Length conditioning**. Captioning with length hints is a superior pre-training task compared to captioning alone as evident by the improved captioning and downstream results.

2×2 pooling	cap F_1	11-avg
stacking	53.7	76.1
attention	**54.1**	**76.9**

(f) **Pooling**. Pooling 2×2 windows of vision tokens using mean-query attention performs better than simply stacking the four features as input to the vision-language connector MLP.

Table 2. **Model ablations.** Default settings are marked in gray. See the Appendix for additional ablations.

Georgia Tech

Stage-3:- The Training Phase

Part-A: - Pre - Training

What are the technical details related to pre-training MOLMO ?

Input Data	The **PixMo-Cap** dataset — 712 K diverse images, 1.3 M human voice-based transcripts and long captions (≈ 196 words per caption)
Optimizer	AdamW (β = 0.9, 0.95; ε = 1e-6).
Epochs / Steps	~4 epochs over PixMo-Cap (~22 K steps for 7B model).
Precision	Mixed precision (AMP): activations → bfloat16; weights & grad reduce → float32 (for stability).
Parallelizatio n	Fully Sharded Data Parallel (FSDP)
Epochs / Steps	~4 epochs over PixMo-Cap (~22 K steps for 7B model).
Sequence Length	Max 2304 tokens (vision + text).

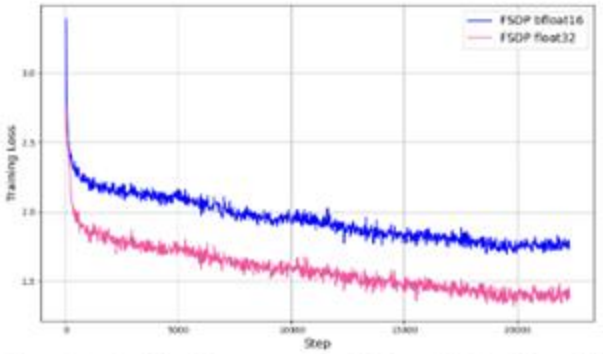


Figure 6. **Training loss curves** for Molmo-7B-D with model weights and gradient reduction in bfloat16 (blue) and float32 (pink). Float32 is our default configuration.

Georgia Tech

Molmo: Pre-Training(Ablations)

- Dataset usage: **Prompts Used:-** Model is prompted with either "long caption:" (for detailed caption) OR "transcript:" (for spoken-style output)
- For images with multiple captions/transcripts: all text tokens are concatenated in one sequence with *attention masks* → each annotation attends only to its own text + image tokens. Saves compute (~ 2 × faster).
- **Length Hint:** Numerical token in prompt controls caption verbosity ("long caption 70:"); Improves recall/precision trade-off.
- **Text-only Dropout:** Drop text tokens to force reliance on visual tokens (better grounding).
- **Connector Fast Warmup:** Higher LR + short warmup → no need for separate connector pre-training, since cleaner data
- **Full FP32 weights + AMP:** Prevents numerical instability at scale.

Part- B: - Post - Training

Georgia Tech

What are the technical details for post tuning ?

Figure 4. Datasets used for fine-tuning, shown in proportion to their sampling rates. Green denotes human-annotated data we collected, blue denotes synthetic data we generated, and purple represents pre-existing academic datasets. PixMo-Docs has been subdivided into charts, tables, diagrams, and other.

Input Data	Combines PixMo (AskModelAnything, Points, Count, Docs, Clocks, CapQA) + 15+ academic datasets (VQA, ChartQA, DocVQA, etc.).
Optimizer	AdamW (β = 0.9, 0.95; ε = 1e-6).
Epochs / Steps	~4 epochs over PixMo-Cap (~22 K steps for 7B model).
Precision	Mixed precision (AMP): activations → bfloat16; weights & grad reduce → float32 (for stability).
Parallelization	Fully Sharded Data Parallel (FSDP)
Sampling rule: -	Proportional to √(dataset size) to avoid dominance by large synthetic sets.Pointing data heavily up-weighted.

Georgia Tech

What are some other fine-tuning strategies?

Problem: - When fine-tuning on 15+ different datasets (VQA, DocVQA, ChartQA, PixMo-Points, etc.), each dataset has different answer styles, different output formats, and different question tones. This was not done for Pixmo datasets !

If you train them together without separation: The model might confuse formats (e.g., answering a chart question like a VQA question), or lose conversational tone because benchmark answers are short and mechanical.

Solution → Introduce lightweight text prefixes ("style tags"). These are short tokens inserted at the start of the input prompt, telling the model what kind of data/task this example belongs

Dataset:	Example Input
VQA v2.0	vqa2: What is the man holding?
TextVQA	textvqa: What does the sign say?

When Fine-tuning:-

Input sequence (simplified)

```
[IMG_START] ...vision tokens... [IMG_END]
"chartqa:" "What" "was" "the" "sales" ... "?"
→ model predicts "The", "sales", "were", "10",
"billion", "."
```

For pointing:

```
<point x="42.3" y="55.1" alt="dog">dog</point>
```

Model learns to *chain-of-thought count* by pointing sequentially.

Georgia Tech

What are the key details from both the training phases ?

- All components (ViT, Connector, LLM) remain **trainable**, but with smaller LRs(during fine tuning) and higher LRs(during pre training)
- FSDP + AdamW + cosine decay (same setup) for pre and post training

	pre-train			fine-tune		
	GPUs	time	GPU hr.	GPUs	time	GPU hr.
1B-E	8	33.3	264	64	13.3	850
7B-D	64	8.6	550	128	11.2	1.4k
7B-O	64	8.9	570	128	13.5	1.7k
72B	128	33.3	4.2k	256	32.4	8.3k

Table 8. Training times for the Molmo models using H100 GPUs.

Stage	Purpose	Data	Key Tricks	Output
Pre-Training	Align vision & language	PixMo-Cap (dense captions)	Length hints, overlap crops, text-only dropout, connector fast warmup	Generates detailed image captions
Fine-Tuning	Teach reasoning & instruction following	PixMo + academic datasets	Style tags, multi-task batching, up-weight pointing	Answers, counts, and points to objects

Any Questions ?

Evaluation

What is the point of that Benchmark ?

#	Benchmark	What It Is	Skill Tested	Example Question
1	**AI2D — Science Diagrams**	Multiple-choice questions about science diagrams (arrows, labels, parts, flows).	Diagram reading; spatial and semantic relations	“Which arrow shows heat flow?”
2	**ChartQA — Charts & Plots**	Question answering over bar, line, and pie charts.	OCR; numerical reading; basic arithmetic/aggregation	“What is the 2019 sales for Europe?”
3	**VQA v2.0 — Everyday Photos**	Visual question answering on natural images with short answers.	General visual understanding; commonsense reasoning	“What color is the bus?”
4	**DocVQA — Documents (Scans, Forms)**	QA on document images such as forms, receipts, and pages.	OCR; layout and document structure understanding	“What is the total due?”
5	**InfoQA — Infographics**	QA over infographic-style visuals mixing text and images.	OCR; reasoning over mixed text and visual elements	“According to the infographic, which country leads in X?”
6	**TextVQA — Reading Text in the Wild**	QA on natural photos where recognizing text is essential.	Scene text detection, recognition, and grounding	“What does the street sign say?”

What is the point of that Benchmark(contd.) ?

#	Benchmark	What It Is	Skill Tested	Example Question
7	**RealWorldQA — Zero-shot Natural Photos**	QA on diverse, real-world images unseen in training.	Zero-shot generalization; robust visual understanding	“Is the person wearing a helmet?”
8	**MMMU — Multi-Domain Reasoning**	Academic-style reasoning tasks across many subjects.	Multi-step reasoning with images	“Given the labeled circuit, which bulb is brightest?”
9	**MathVista — Visual Math Reasoning**	Math problems involving visual diagrams or figures.	Math reasoning; geometry; multi-step logic	“What is the angle at point B?”
10	**CountBenchQA — Counting in Images**	Counting objects in natural or cluttered scenes.	Object counting; grounding	“How many red chairs are there?”
11	**PixMo-Count — Hard Counting**	A more difficult counting benchmark with messy, real scenes.	Robust counting; localization under noise	“Count the people wearing helmets.”
12	**Human Preference (Elo)**	Human evaluation via pairwise preference comparisons (~15k prompts, ~870 raters).	Overall multimodal answer quality and alignment with human judgment	“Which model gave a better explanation?”

model	AI2D test [49]	ChartQA test [82]	VQA v2.0 testdev [36]	DocVQA test [83]	InfoQA test [84]	TextVQA val [100]	RealWorldQA [116]	MMMU val [129]	MathVista testmini [78]	CountBenchQA [10]	PixMo-Count test	Average	Elo score	Elo rank
API call only														
GPT-4V [88]	89.4	78.1	77.2	87.2	75.1	78.0	61.4	63.1	58.1	69.9	45.0	71.1	1041	10
GPT-4o-0513 [90]	94.2	85.7	78.7	92.8	79.2	77.4	75.4	**69.1**	63.8	87.9	59.6	78.5	**1079**	**1**
Gemini 1.5 Flash [103]	91.7	85.4	80.1	89.9	75.3	78.7	67.5	56.1	58.4	81.6	61.1	75.1	1054	7
Gemini 1.5 Pro [103]	94.4	87.2	80.2	93.1	81.0	78.7	70.4	62.2	63.9	85.8	64.3	78.3	1074	3
Claude-3 Haiku [7]	86.7	81.7	68.4	88.8	56.1	67.3	45.5	50.2	46.4	83.0	43.9	65.3	999	18
Claude-3 Opus [7]	88.1	80.8	66.3	89.3	55.6	67.5	49.8	59.4	50.5	83.6	43.3	66.7	971	21
Claude-3.5 Sonnet [7]	94.7	**90.8**	70.7	95.2	74.3	74.1	60.1	68.3	**67.7**	89.7	58.3	76.7	1069	4
Open weights only														
PaliGemma-mix-3B [10]	72.3	33.7	76.3	31.3	21.4	56.0	55.2	34.9	28.7	80.6	60.0	50.0	937	27
Phi3.5-Vision-4B [1]	78.1	81.8	75.7	69.3	36.6	72.0	53.6	43.0	43.9	64.6	38.3	59.7	982	19
Qwen2-VL-7B [111]	83.0	83.0	82.9	94.5	76.5	84.3	70.1	54.1	58.2	76.5	48.0	73.7	1025	14
Qwen2-VL-72B [111]	88.1	88.3	81.9	**96.5**	84.5	**85.5**	**77.8**	64.5	70.5	80.4	55.7	79.4	1037	12
InternVL2-8B [104]	83.8	83.3	76.7	91.6	74.8	77.4	64.2	51.2	58.3	57.8	43.9	69.4	953	23
InternVL2-Llama-3-76B [104]	87.6	88.4	85.6	94.1	**82.0**	84.4	72.7	58.2	65.5	74.7	54.6	77.1	1018	16
Pixtral-12B [3]	79.0	81.8	80.2	90.7	50.8	75.7	65.4	52.5	58.0	78.8	51.7	69.5	1016	17
Llama-3.2V-11B-Instruct [5]	91.1	83.4	75.2	88.4	63.6	79.7	64.1	50.7	51.5	73.1	47.4	69.8	1040	11
Llama-3.2V-90B-Instruct [5]	92.3	85.5	78.1	90.1	67.2	82.3	69.8	60.3	57.3	78.5	58.5	74.5	1063	5
Open weights + data († distilled)														
LLaVA-1.5-7B [69]	55.5	17.8	78.5	28.1	25.8	58.2	54.8	35.7	25.6	40.1	27.6	40.7	951	26
LLaVA-1.5-13B [69]	61.1	18.2	80.0	30.3	29.4	61.3	55.3	37.0	27.7	47.1	35.2	43.9	960	22
xGen-MM-interleave-4B† [119]	74.2	60.0	81.5	61.4	31.5	71.0	61.2	41.1	40.5	81.9	50.2	59.5	979	20
Cambrian-1-8B† [106]	73.0	73.3	81.2	77.8	41.6	71.7	64.2	42.7	49.0	76.4	46.6	63.4	952	25
Cambrian-1-34B† [106]	79.7	75.6	83.8	75.5	46.0	76.7	67.8	49.7	53.2	75.6	50.7	66.8	953	24
LLaVA OneVision-7B† [59]	81.4	80.0	84.0	87.5	68.8	78.3	66.3	48.8	63.2	78.8	54.4	72.0	1024	15
LLaVA OneVision-72B† [59]	85.6	83.7	85.2	91.3	74.9	80.5	71.9	56.8	67.5	84.3	60.7	76.6	1051	8
The Molmo family: Open weights, Open data, Open training code, Open evaluations														
MolmoE-1B	86.4	78.0	83.9	77.7	53.9	78.8	60.4	34.9	34.0	87.2	79.6	68.6	1032	13
Molmo-7B-O	90.7	80.4	85.3	90.8	70.0	80.4	67.5	39.3	44.5	89.0	83.3	74.6	1051	9
Molmo-7B-D	93.2	84.1	85.6	92.2	72.6	81.7	70.7	45.3	51.6	88.5	84.8	77.3	1056	6
Molmo-72B	**96.3**	87.3	**86.5**	93.5	81.9	83.1	75.2	54.1	58.6	**91.2**	**85.2**	**81.2**	1077	2

What did we achieve ?

Table 1. We present academic benchmark results for 10 common datasets, plus a new counting benchmark, PixMo-Count, which features more challenging natural images than CountBenchQA. We categorize models into four groups: (top) proprietary models accessible only via API calls, (upper middle) models with released weights but closed data, (lower middle) models with released weights and training data (noting some of these use distillation (†) from proprietary VLMs via synthetic data), and (bottom) the Molmo family of models.

Georgia Tech

What is the conclusion of these results ?

Overall Performance

- **Molmo-72B** ranks **#2 overall** (just behind GPT-4o) → Beats **Gemini 1.5 Pro**, **Gemini 1.5 Flash**, and **Claude 3.5 Sonnet**.**(Elo ranking)**
- **Molmo-7B** and **MolmoE-1B** models perform between **GPT-4V** and **GPT-4o** while being fully open.
- Achieves **state-of-the-art among open models** — and all weights, data, and code are released.

Where Molmo Excels

- **Visual Understanding & Captioning :-** Excellent at describing complex natural images; ranks top on these benchmarks.
- **Counting & Grounding: -** Best-in-class due to new point-then-count reasoning and 2D pointing data.
- **Diagram & Chart Interpretation:-** Performs near top; overlapping multi-crops preserve fine visual details.
- **Document & OCR Tasks:-** After multimodal training, a small drop in text-only skills (recovered by fine-tuning with Tulu-3).

Average / Needs Improvement

- **Reasoning & Math :-** Weaker reasoning and math logic; model not trained with enough structured reasoning data.
- **Fine OCR & Text-heavy Scenes:-** Slightly behind **Qwen2-VL**, which is heavily optimized for OCR.
- **Text-Only Knowledge / Coding:-** After multimodal training, a small drop in text-only skills (recovered by fine-tuning with Tulu-3).

Georgia Tech

Other Results: - CHATBOT ARENA

What it is: Third-party human preference leaderboard (pairwise votes → Elo).

What Molmo did:

- **Molmo-72B** beats **all fully open/open-weight** models there, but sits **below top proprietary** models.
- In Molmo's **own** controlled Elo study (Section 5), **Molmo-72B ranks #2 overall** (just behind GPT-4o).

model	score	95% CI	opennness
Gemini-Exp-114 [103]	1278	+28/-27	API only
ChatGPT-4o-latest (20240903) [90]	1256	+13/-13	API only
Gemini-1.5-Pro-002 [103]	1220	+15/-14	API only
Gemini-1.5-Flash-002 [103]	1219	+15/-17	API only
GPT-4o-2024-05-13 [90]	1213	+9/-9	API only
Claude 3.5 Sonnet (20240620) [7]	1187	+9/-7	API only
Claude 3.5 Sonnet (20241022) [7]	1184	+15/-15	API only
Gemini-1.5-Pro-001 [103]	1158	+9/-8	API only
GPT-4-Turbo-2024-04-09 [88]	1157	+7/-10	API only
Gemini-1.5-Flash-8B-Exp-0827 [103]	1137	+15/-13	API only
GPT-4o-2024-08-06 [90]	1131	+18/-20	API only
Gemini-1.5-Flash-8B-001 [103]	1133	+10/-15	API only
GPT-4o-mini-2024-07-18 [89]	1124	+7/-9	API only
Molmo-72B	1115	+18/-17	Fully Open
Qwen2-VL-72B [111]	1113	+15/-17	Open Weight
InternVL2-26B [104]	1096	+11/-10	Open Weight
Pixtral-12B-2409 [3]	1085	+13/-14	Open Weight
Llama-3.2V-90B-Instruct [5]	1085	+12/-14	Open Weight
Gemini-1.5-Flash-001 [103]	1087	+8/-8	API only
Molmo-7B-D	1076	+15/-18	Fully Open
Yi-Vision [4]	1070	+21/-26	Distilled
Claude 3 Opus [7]	1073	+6/-8	API only
Qwen2-VL-7B [111]	1068	+15/-14	Open Weight
Llama-3.2V-11B-Instruct [5]	1061	+14/-14	Open Weight

Table 9. **Chatbot Arena's vision leaderboard** for English queries. The table is up to date as of Nov. 13, 2024. We show up to 20 rows for clarity.

Georgia Tech

Other Results: - CLOCK Reading

- **Setup:** Train on **synthetic watch faces** (PixMo-Clocks), test **in the wild** (COCO, OpenImages, 'Clock Movies').
- **Prompt:** "What time is being shown? Answer as HH:MM."
- **Result:** Most VLMs—open and closed—**struggle**.
- **Molmo models dominate** VLMs (overall/hour/minute accuracy), though a **specialized single-task clock model** still wins.

model	acc.	hour acc.	min. acc.
GPT-4o-0513 [90]	2.7	14.2	8.6
Gemini 1.5 Pro [103]	0.9	11.6	5.1
Claude-3.5 Sonnet [7]	6.6	22.3	17.5
PaliGemma-mix-3B [10]	6.1	21.0	15.8
Phi3.5-Vision-4B [1]	1.9	12.0	7.6
Qwen2-VL-72B [111]	9.1	24.9	18.4
InternVL2-Llama-3-76B [104]	3.3	16.3	9.9
Pixtral-12B [3]	1.7	11.9	6.7
Llama-3.2V-90B-Instruct [5]	3.4	17.9	10.1
LLaVA-1.5-13B [69]	0.8	11.6	5.7
xGen-MM-interleave-4B [119]	2.0	11.9	8.0
Cambrian-1-34B [106]	1.8	11.1	7.2
LLaVA OneVision-72B [59]	5.7	17.9	15.4
MolmoE-1B	65.8	77.9	74.1
Molmo-7B-O	64.2	76.3	73.8
Molmo-7B-D	**68.2**	**78.6**	**76.0**
Molmo-72B	65.6	77.1	73.7
Specialized single-task model [121]	78.9	84.2	82.9

Table 10. **Clock reading benchmark results**. We report the averages of overall, hour and minute accuracies, each evaluated on three different test sets based on COCO, OpenImages and Clock Movies, respectively. **Bold** numbers represent the highest VLM scores while the best numbers, excluding Molmo, are underlined. We categorize models into five groups: (first) API-only, (second) open-weight, (third) open-weight and open-data, (four) the Molmo family and (five) the specialized clock reading model.

Georgia Tech

Conclusion

What is the conclusion from all this ?

Molmo set out to prove that multimodal reasoning can be achieved openly — with transparent data, modular architecture, and reproducible training recipes.

Key Contributions: -

- **PixMo Dataset:** High-quality, LLM-assisted but auditable multimodal data — bridging web-scale diversity with detailed grounding (captions, points, documents, clocks, counts).
- **Molmo Model:** Simple yet powerful architecture — multiscale overlapping crops + attention pooling connector + open LLM — that achieves competitive reasoning without closed data.
- **Openness:** Every stage — data, code, checkpoints, evaluation — is public and reproducible, setting a new standard for transparency in VLMs.

Quick Demo !

Discussion

Where Do We Go From Here?

Q1:- PixMo introduces separate datasets for every new capability (counting, clock reading, document QA). Do we risk fragmenting ‘intelligence’ into narrow subskills instead of achieving general reasoning?

Q-2:- If data diversity matters more than sheer scale, what does an ideal next-generation multimodal dataset look like — curated, synthetic, or mixed?

Q-3:- With all the VLM architecture seen, can we conclude now that if we combine techniques we will get the best model ?

Q-4:- While training how much emphasis to text v/s image(dropout layer in MOLMO’s LLM) ?

Q-5:- Is data still the bottleneck — or is the current problem in our architecture or context for models?

Q-6:- As VLMs evolve toward multimodal agents (seeing, hearing, acting), what defines true intelligence — performance on datasets, or the ability to generalize without new data?

Q-7:- Papers like Imagebind, Unified-IO-2, combine modalities under a shared token space, does that mark the end of modular encoders and connectors like in Molmo — or will modularity remain important for specialization?

Georgia Tech