Chameleon, Janus, and Janus Pro

Presented by Mohit Talreja, Saikrishnan Sankar, and Jakob Bjorner

Georgia Tech

Before the beginning there was Chameleon

Unified generation and image understanding with image tokens

Chameleon: Mixed-Modal Early-Fusion Foundation Models

Chameleon Team[1,*]

[1]FAIR at Meta
*See Contributions section for full author list.

We present Chameleon, a family of early-fusion token-based mixed-modal models capable of understanding and generating images and text in any arbitrary sequence. We outline a stable training approach from inception, an alignment recipe, and an architectural parameterization tailored for the early-fusion, token-based, mixed-modal setting. The models are evaluated on a comprehensive range of tasks, including visual question answering, image captioning, text generation, image generation, and long-form mixed modal generation. Chameleon demonstrates broad and general capabilities, including state-of-the-art performance in image captioning tasks, outperforms Llama-2 in text-only tasks while being competitive with models such as Mixtral 8x7B and Gemini-Pro, and performs non-trivial image generation, all in a single model. It also matches or exceeds the performance of much larger models, including Gemini Pro and GPT-4V, according to human judgments on a new long-form mixed-modal generation evaluation, where either the prompt or outputs contain mixed sequences of both images and text. Chameleon marks a significant step forward in a unified modeling of full multimodal documents.

Date: May 17, 2024

Prompt: <img> While chameleons are masters of camouflage in the natural world, let's see how they might fare in a man-made setting.
1. Analyze the colors, patterns, and textures in this image and identify the most challenging spot for a chameleon to blend in.
2. Then generate an image of a chameleon with the optimal pattern to camouflage itself.
3. Then, rate and explain how successful each attempt is on a scale from 1-10.

Response: The image is of a stop sign in a forested area, with some sunlight filtering through the trees. The sign has a red octagonal shape with a white border and the word "STOP" written in white letters. In the background, there are green leaves and trees.

Challenging Spot for a Chameleon: The stop sign would be the most challenging spot for a chameleon to blend in due to its bright red color and bold white letters, which would stand out in the forested environment.

Optimal Pattern for Camouflage: To camouflage itself on the stop sign, a chameleon would need to have a pattern that would break up its outline and make it less conspicuous. A pattern with shades of brown and red would help the chameleon blend in more effectively. Additionally, a mottled or dappled pattern would help to scatter light and further reduce the chameleon's visibility.

<img>
Rating and Explanation: A rating of 2/10 would be given for the chameleon's ability to blend in on the stop sign. While the chameleon's green and brown pattern helps to break up its outline, the bright red color of the stop sign will still make it stand out in the forested environment.

Chameleon Diagram

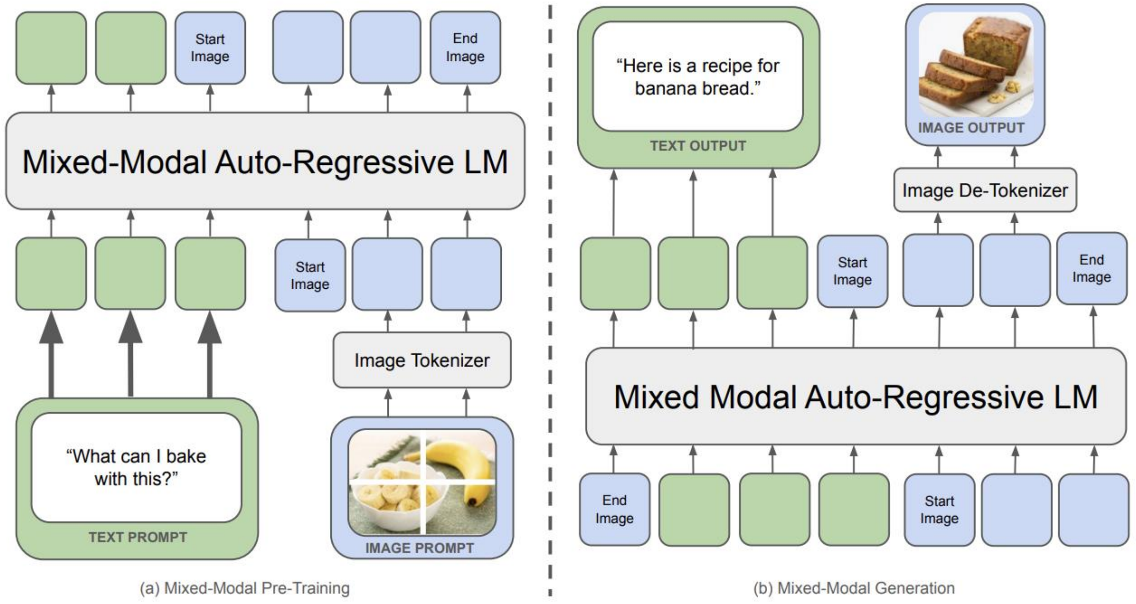


Figure 1 Chameleon represents all modalities — images, text, and code, as discrete tokens and uses a uniform transformer-based architecture that is trained from scratch in an end-to-end fashion on ~10T tokens of interleaved mixed-modal data. As a result, Chameleon can both reason over, as well as generate, arbitrary mixed-modal documents. Text tokens are represented in green and image tokens are represented in blue.

Pre-normalization [GPT3]. To improve the training stability, we normalize the input of each transformer sub-layer, instead of normalizing the output. We use the RMSNorm normalizing function, introduced by Zhang and Sennrich (2019).

SwiGLU activation function [PaLM]. We replace the ReLU non-linearity by the SwiGLU activation function, introduced by Shazeer (2020) to improve the performance. We use a dimension of $\frac{2}{3}4d$ instead of $4d$ as in PaLM.

Rotary Embeddings [GPTNeo]. We remove the absolute positional embeddings, and instead, add rotary positional embeddings (RoPE), introduced by Su et al. (2021), at each layer of the network.

The details of the hyper-parameters for our different models are given in Table 2.

Chameleon Datasets

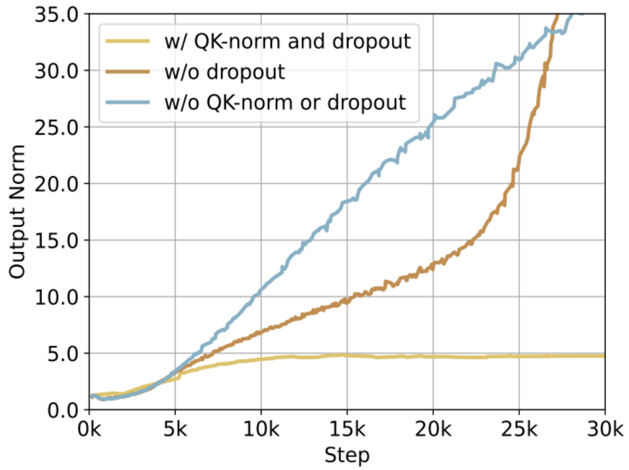
Data domain	Source	Number tokens
Text Only	LLaMa-2 and CodeLLaMa data.	2.9 trillion text-only tokens
Text-Image	Publicly and licensed data. 512 × 512 images for tokenization.	1.4 billion text-image pairs = 1.5 trillion text-image tokens
Text/Image Interleaved	Publicly available web sources for	400 billion tokens of interleaved text and image data Filtered for quality

Chameleon's Plague

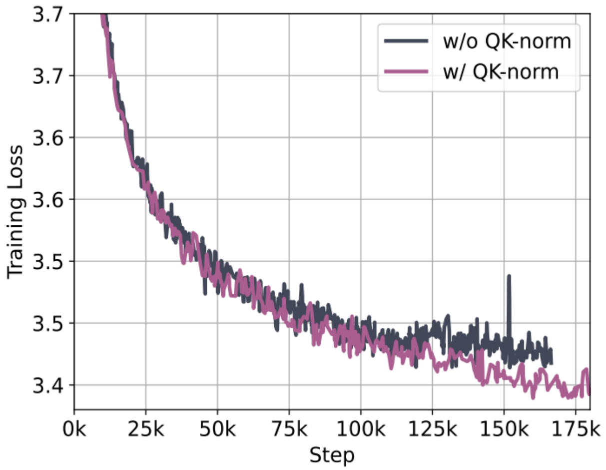
Plagued with instability due to logit competition in softmax

$$softmax(z) = softmax(z + c)$$

applying layer norm to the query and key vectors within the attention.



(a) Uncontrolled growth of output norms is a strong indicator of future training divergence.



(b) An ablation with Chameleon-7B with and without QK-Norm.

Chameleon's Plague

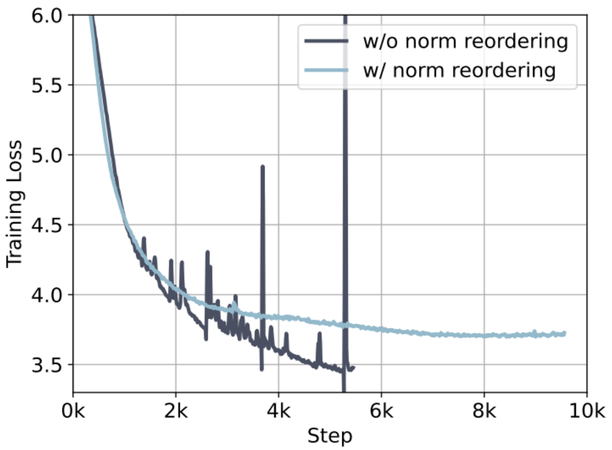
Chameleon-34B: $h = x + \text{attention\_norm}(\text{attention}(x))$

$\text{output} = h + \text{ffn\_norm}(\text{feed\_forward}(h))$

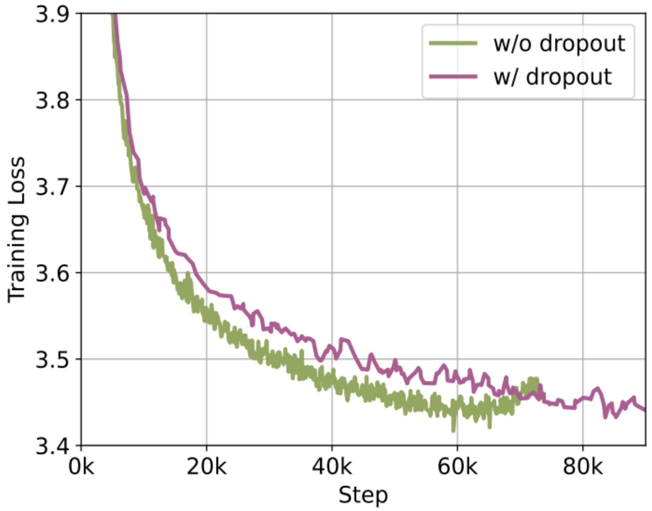
Llama2: $h = x + \text{attention}(\text{attention\_norm}(x))$

$\text{output} = h + \text{feed\_forward}(\text{ffn\_norm}(h))$

we apply **z-loss** regularization. Specifically, we regularize the partition function Z of the softmax function $\sigma(x)_i = \frac{e^{x_i}}{Z}$ where $Z = \sum_i e^{x_i}$ by adding $10^{-5} \log^2 Z$ to our loss function.



(c) For Chameleon-34B, using **dropout** does not fix divergences, both with and without norm-reordering.



(c) An ablation with Chameleon-7B with and without **dropout**.

Georgia Tech

Data Dependent control flow

Autoregressive, mixed-modal generation introduces unique performance-related challenges at inference time. These include:

- **Data-dependencies per-step** — given that our decoding formulation changes depending on whether the model is generating images or text at a particular step, tokens must be inspected at each step (i.e. copied from the GPU to the CPU in a blocking fashion) to guide control flow.
- **Masking for modality-constrained generation** — to facilitate exclusive generation for a particular modality (e.g. image-only generation), tokens that do not fall in a particular modality space must be masked and ignored when de-tokenizing.
- **Fixed-sized text units** — unlike text-only generation, which is inherently variable-length, token-based image generation produces fixed-size blocks of tokens corresponding to an image.

each output token must be inspected for image-start tokens to condition image-specific decoding

Chameleon does good but not great on multi modal and unimodal tasks

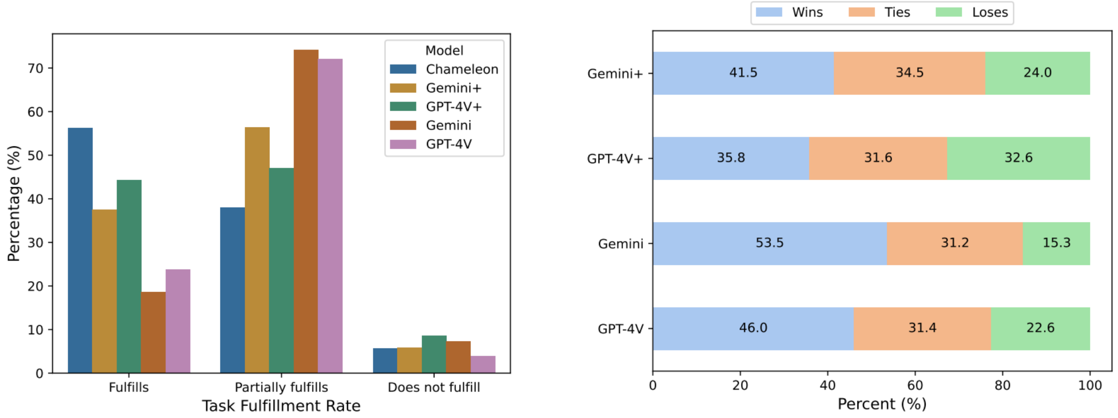
Table 6 Comparison of overall performance on collective academic benchmarks against open-source foundational models. * Evaluated using our framework/using API. For GSM8k/MATH, we report maj@1 unless mentioned otherwise. ** From Gemini et al. (2023).

	Chameleon		Llama-2			Mistral		Gemini Pro	GPT-4
	7B	34B	7B	34B	70B	7B	8x7B	—	—
Commonsense Reasoning and Reading Comprehension									
PIQA	79.6	83.3	78.8	81.9	82.8	83.0	83.6	—	—
SIQA	57.0	63.3	48.3	50.9	50.7	—	—	—	—
HellaSwag	74.2	82.7	77.2	83.3	85.3	81.3	84.4	—	—
	75.6 10-shot	85.1 10-shot	—	—	87.1 10-shot	83.9 10-shot	86.7 10-shot	84.7 10-shot	95.3 10-shot
WinoGrande	70.4	78.5	69.2	76.7	80.2	75.3	77.2	—	—
Arc-E	76.1	84.1	75.2	79.4	80.2	80.0	83.1	—	—
Arc-C	46.5	59.7	45.9	54.5	57.4	55.5	59.7	—	—
OBQA	51.0	54.0	58.6	58.2	60.2	—	—	—	—
BoolQ	81.4	86.0	77.4	83.7	85.0	84.7*	—	—	—
Math and World Knowledge									
GSM8k	41.6	61.4	14.6	42.2	56.8	52.1 maj@8	74.4 maj@8	86.5 maj@32 CoT	92.0 SFT CoT
	50.9 maj@8	77.0 maj@32	—	—	—	—	75.1* maj@32	—	—
MATH	11.5 maj@1	22.5 maj@1	2.5	6.24	13.5	13.1 maj@4	28.4 maj@4	32.6	52.9**
	12.9 maj@4	24.7 maj@4	—	—	—	—	—	—	—
MMLU	52.1	65.8	45.3	62.6	68.9	60.1	70.6	71.8	86.4

Table 7 Model Performances on Image-to-Text Capabilities. * Evaluated using API.

	Model	Model Size	COCO	Flickr30k	VQAv2
Pre-trained	**Flamingo-80B**	80B	113.8 32-shot	75.1 4-shot	67.6 32-shot
	IDEFICS-80B	80B	116.6 32-shot	73.7 4-shot	65.9 32-shot
Chameleon	**Chameleon**	34B	120.2 2-shot	74.7 2-shot	66.0 2-shot
	Chameleon-SFT	34B	140.8 0-shot	82.3 2-shot	—
	Chameleon-MultiTask	34B	139.1 2-shot	76.2 2-shot	69.6
Fine-tuned	**Flamingo-80B-FT**	80B	138.1	—	82.0
	IDEFICS-80B-Instruct	80B	123.2 32-shot	78.4 32-shot	68.8 32-shot
Closed Source (finetuning status unknown)	**GPT-4V**	—	78.5* 8-shot	55.3* 8-shot	77.2
	Gemini Nano 2	—	—	—	67.5
	Gemini Pro	—	99.8* 2-shot	82.2* 4-shot	71.2
	Gemini Ultra	—	—	—	77.8

Outperformed baselines on human evaluations



(a) The prompt task fulfillment rates.

(b) Chameleon vs. the baselines: Gemini+, GPT-4V+, Gemini, GPT-4V.

Figure 9 Performance of Chameleon vs baselines, on mixed-modal understanding and generation on a set of diverse and natural prompts from human annotators.

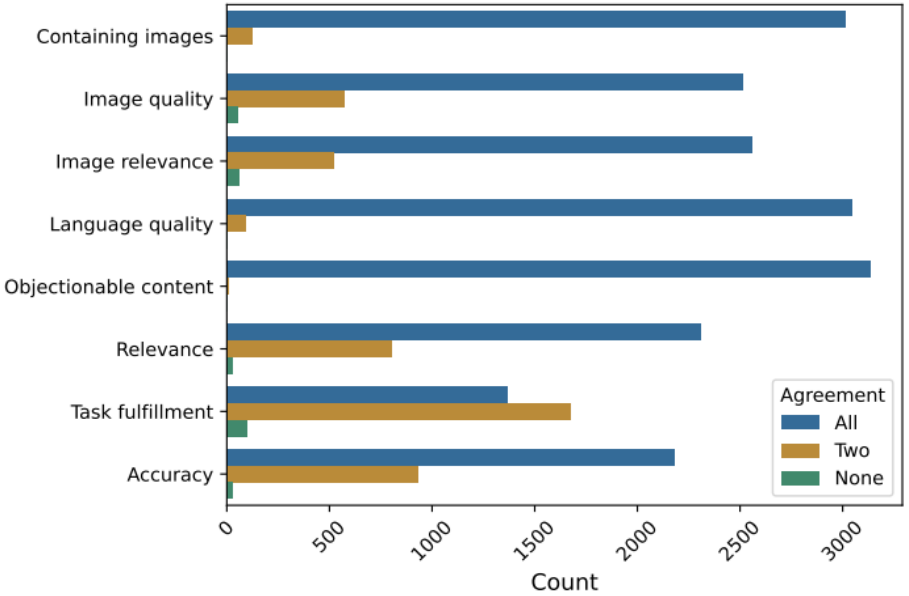


Figure 10 The inter-annotator agreement on the questions in the absolute evaluation.

Janus: Decoupling Visual Encoding for Unified Multimodal Understanding and Generation

17th October, 2024

Authors: Chengyue Wu[1,2], Xiaokang Chen[1,†] *(Project lead), Zhiyu Wu[1,3], Yiyang Ma[1,3], Xingchao Liu[1], Zizheng Pan[1], Wen Liu[1], Zhenda Xie[1], Xingkai Yu[1], Chong Ruan[1], Ping Luo[2]*
Affiliations: [1] DeepSeek-AI, [2] The University of Hong Kong, [3] Peking University

Georgia Tech

Problem Statement & Motivation

Drawback of Chameleon: Single visual encoder for both understanding and generation tasks leads to suboptimal performance

Key Issues:

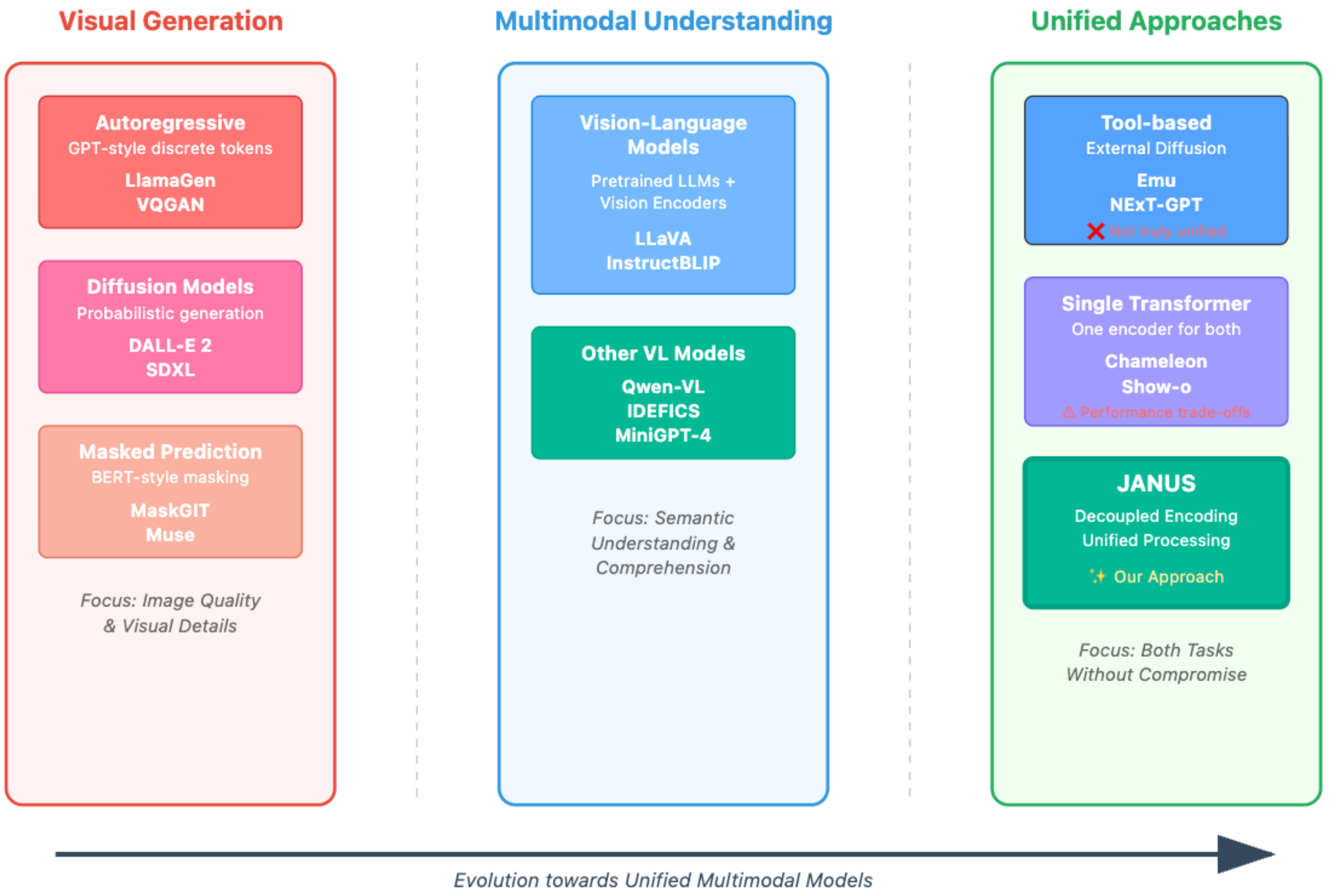
- Understanding needs high-level semantics
- Generation needs low-level details
- Single encoder creates conflicting trade-offs
- Poor multimodal understanding performance

Motivation: Decouple visual encoding to eliminate conflicts while maintaining unified processing

Solution: *Janus uses separate encoders for understanding and generation while processing through a unified transformer architecture*

Why named Janus?

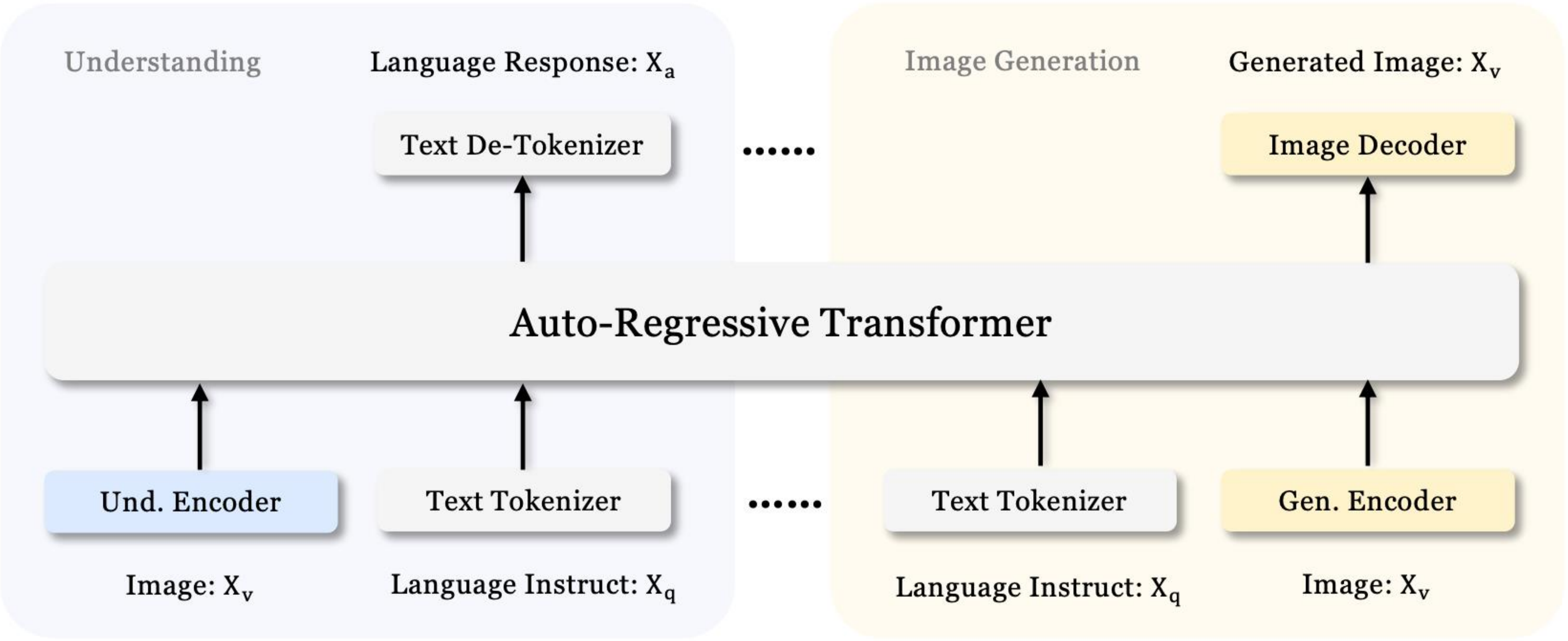
Related Works



Janus : First to explicitly decouple visual encoding while maintaining unified processing architecture

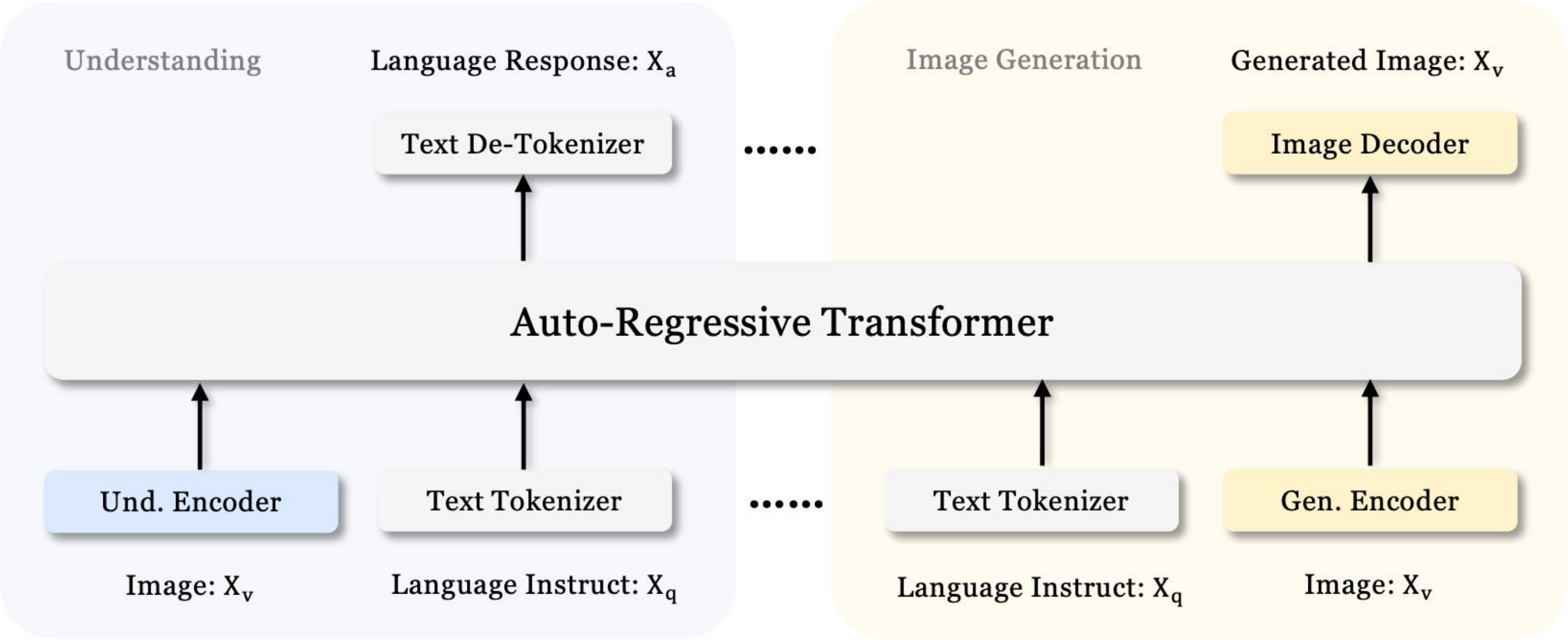
Georgia Tech

Architecture



Georgia Tech

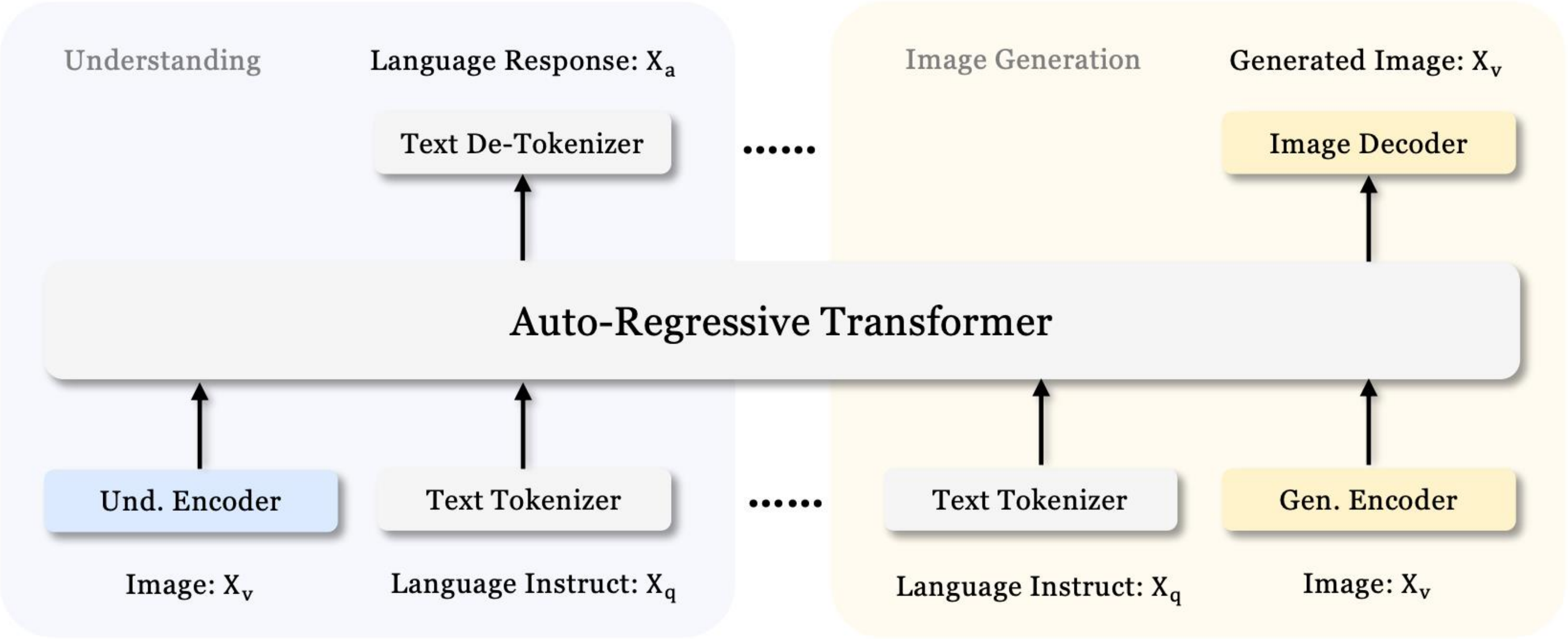
Architecture



Understanding Encoder : SigLIP

- **Input**: Images resized to **384 × 384**, patch size **16 × 16**.
- **Architecture**: Transformer backbone, outputs **24 × 24 = 576 tokens**, flattened and passed through adapter.
- **Encoder Role**: SigLIP captures **high-level semantic features** of the image.
- Pretrained on **~10B image–text pairs** from Google's **WebLI dataset**

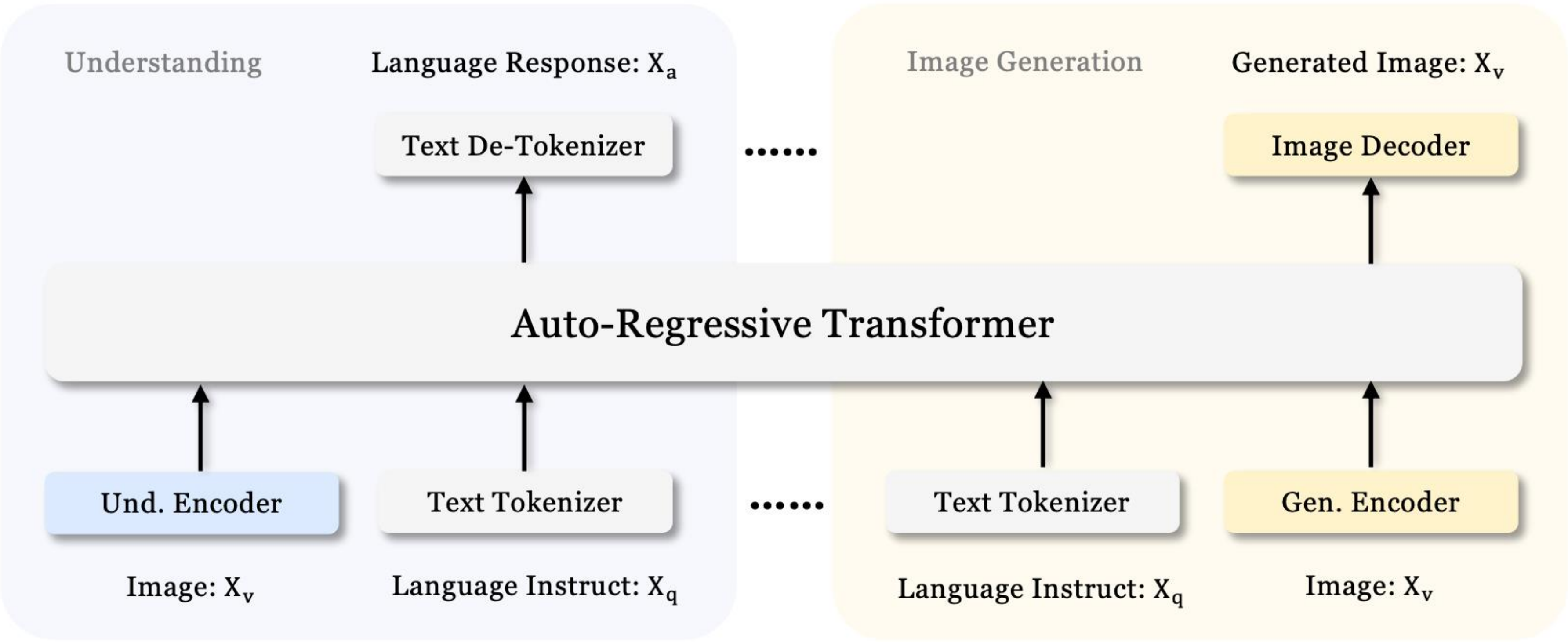
Architecture



Generation Encoder : VQ Tokenizer

- **Input**: Images resized to **384 × 384**, downsampled by a factor of **16**.
- **Architecture**: VQ tokenizer with a **codebook of 16,384** discrete visual tokens.
- **Encoder Role**: Captures **fine-grained spatial and textural details** for image generation.
- **Training Data**: Pretrained with **ImageNet-1k (1.2M images)**

Architecture



Text Tokenizer : BPE Tokenizer (DeepSeek LLM)

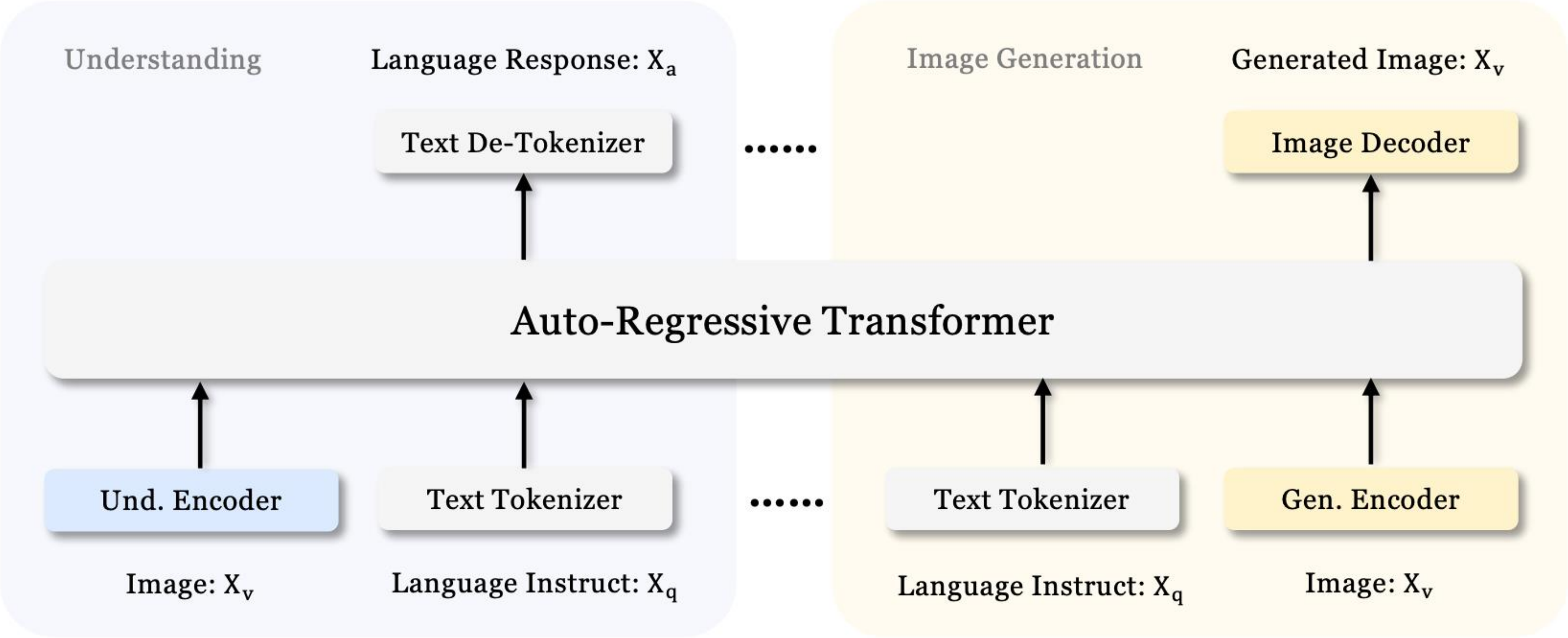
•**Input**: Raw text sequence.

•**Architecture**: Standard **LLM byte-pair tokenizer (BPE)** that converts text into discrete token IDs.

•**Role**: Provides **semantic embeddings** of words/subwords for the LLM; ensures text and image tokens live in the same discrete sequence space.

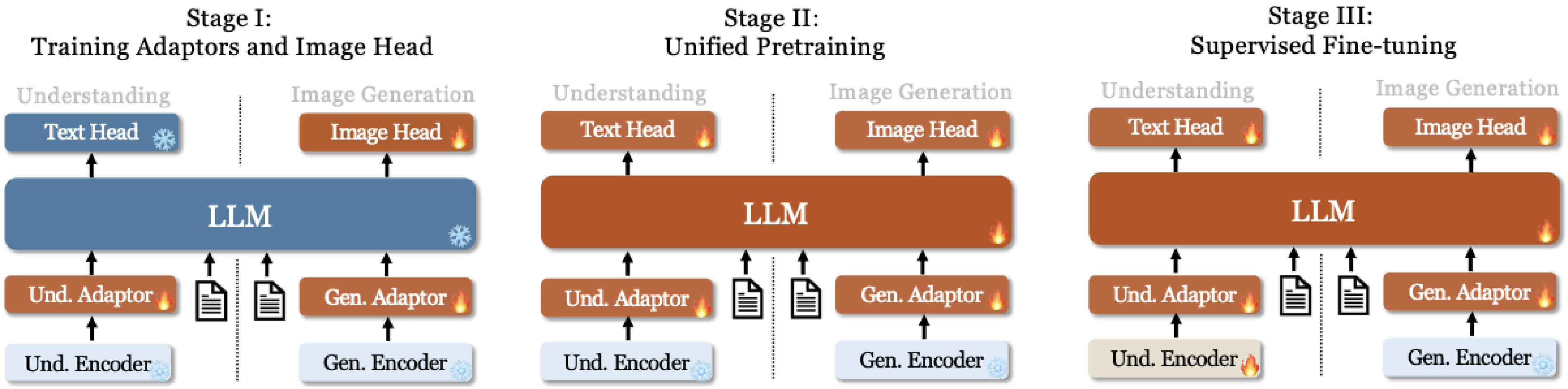
•**Training Data**: Pretrained on **DeepSeek-LLM's text corpus** (hundreds of billions of tokens across web, books, Wikipedia, etc.), before integration into Janus

Architecture



- **LLM Backbone**: **DeepSeek-LLM** (1.3B parameters).
- **Architecture**: Standard **decoder-only transformer**, trained autoregressively (next-token prediction).
- **Sequence Length**: Supports up to **4096 tokens**.
- **Role**: Acts as the **shared multimodal core** – it receives tokens from the text tokenizer, understanding encoder (SigLIP), and generation encoder (VQ tokenizer), then predicts the next token (text or image) in a unified sequence
- Standard cross-entropy loss across all tasks

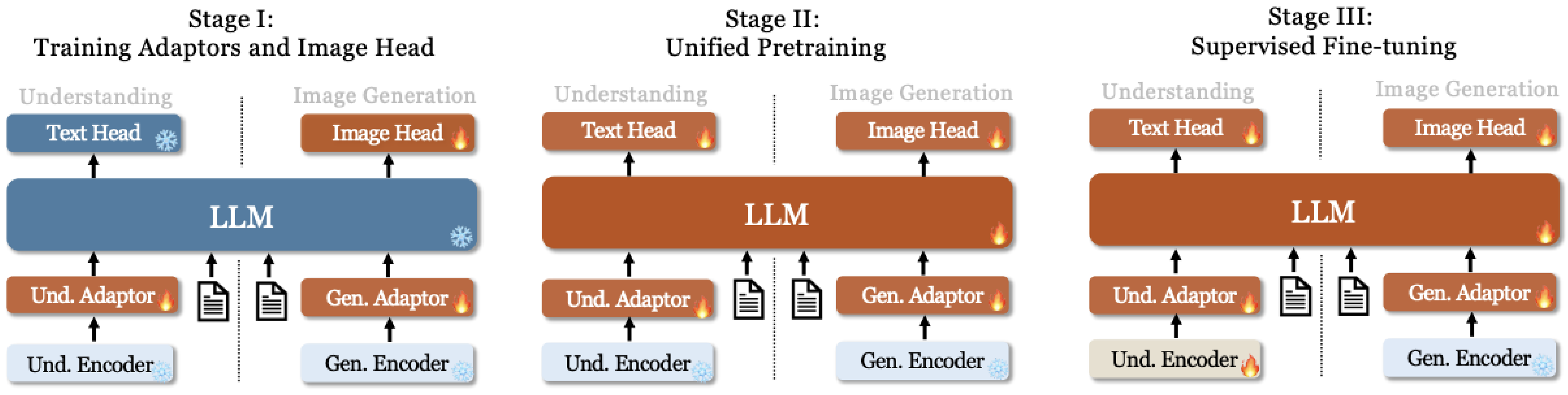
Training Procedure



Adaptors

- **Role**: Lightweight **2-layer MLPs** that map image features (from SigLIP or VQ encoder) into the **LLM token embedding space**.
  - **Input**: High-dim features (SigLIP embeddings / VQ codebook embeddings).[576 Tokens]
  - **Output**: Tokens aligned with the LLM's embedding dimension.

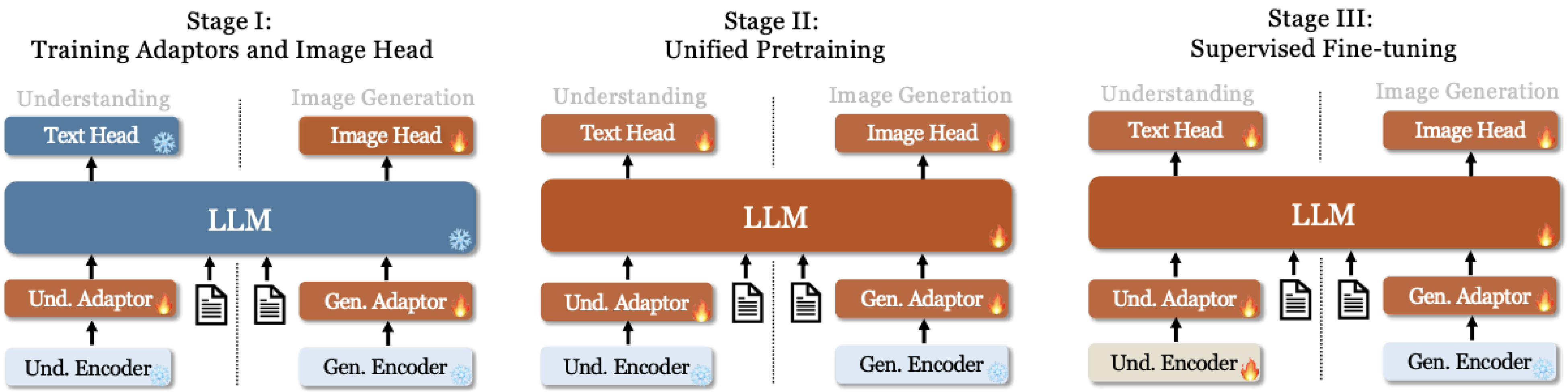
Training Procedure



Stage I: Training Adaptors & Image Head

- **Dataset**:
  - **ShareGPT4V → 1.25M image–text pairs** (for multimodal understanding).
  - **ImageNet-1k → ≈1.2M images** (converted into text-to-image pairs for generation).
- **Size**: **~2.45M samples** total.
- **Goal**: Build a **conceptual bridge** between visual and linguistic features → allow the LLM to begin aligning image embeddings with text tokens, and give it **preliminary image generation ability**, while keeping encoders & LLM **frozen**
- **Data Ratio:** 1 : 0 : 1 (understanding : text-only : Generation)

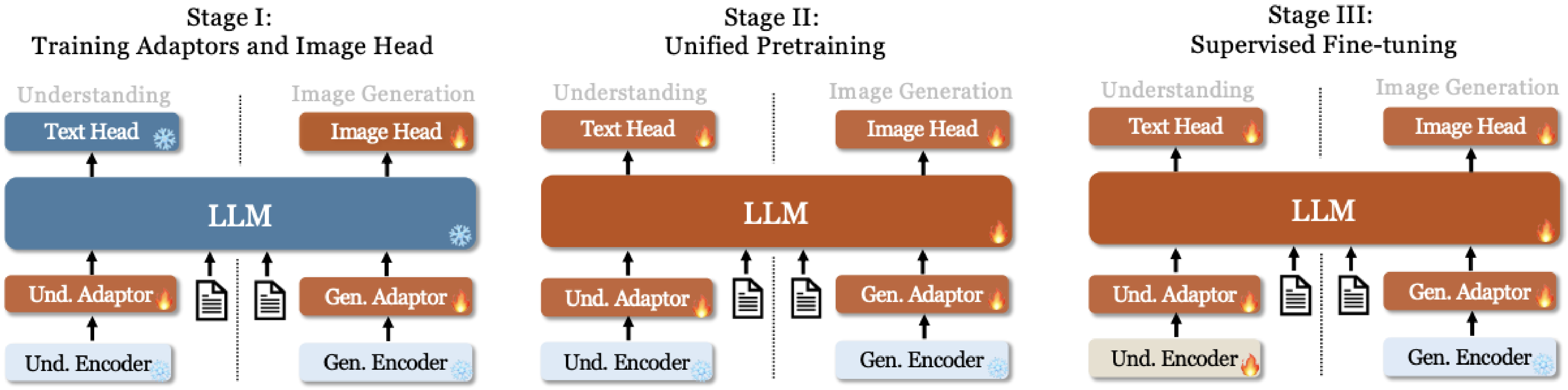
Training Procedure



Stage II: Unified Pretraining

- **Dataset**: Mix of
  - **Text-only corpus** (DeepSeek-LLM pretraining set),
  - **Image–text pairs** (WikiHow, WIT, Open Images, LAION subsets, etc.),
  - **Visual generation data** (~2M in-house caption–image pairs + filtered open-source).
- **Size**: Tens of millions of text + image–text pairs; **ImageNet-1k (1.2M)** used only in early steps.
- **Goal**: Train the model jointly on **text, multimodal understanding, and image generation** → unify modalities under one autoregressive transformer
- **Data Ratio** : 2 : 3 : 5

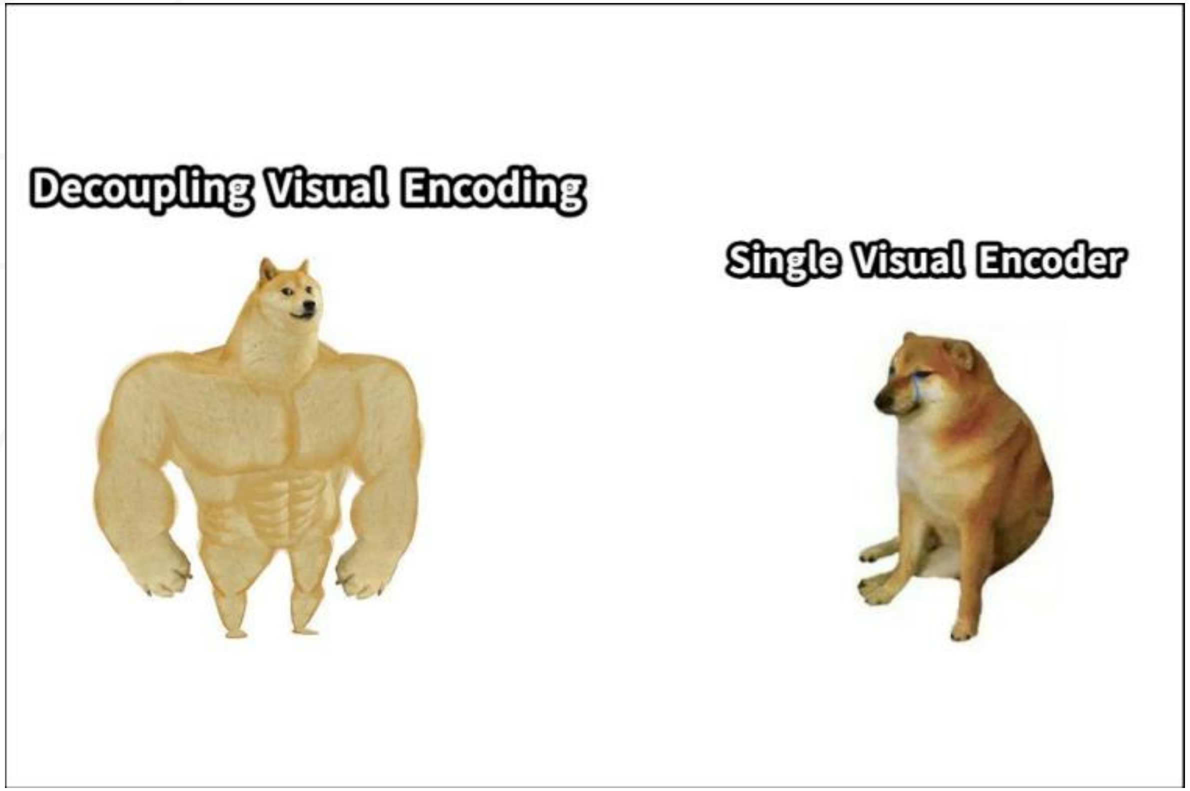
Training Procedure



Stage III: Supervised Fine-tuning

- **Dataset**: Instruction-tuning data:
  - **Text dialogue** corpora,
  - **Multimodal Q&A** (from LLaVA-OneVision, ScreenQA, etc.),
  - **Visual generation instruction pairs** (~4M in-house + open-source).
- **Size**: **Millions of instruction samples** (multi-turn dialogue + image–text pairs).
- **Goal**: Enhance **instruction-following & dialogue ability** across all tasks (text, understanding, generation) while **keeping generation encoder frozen**
- **Data Ratio** : 7 : 3 : 10

Results

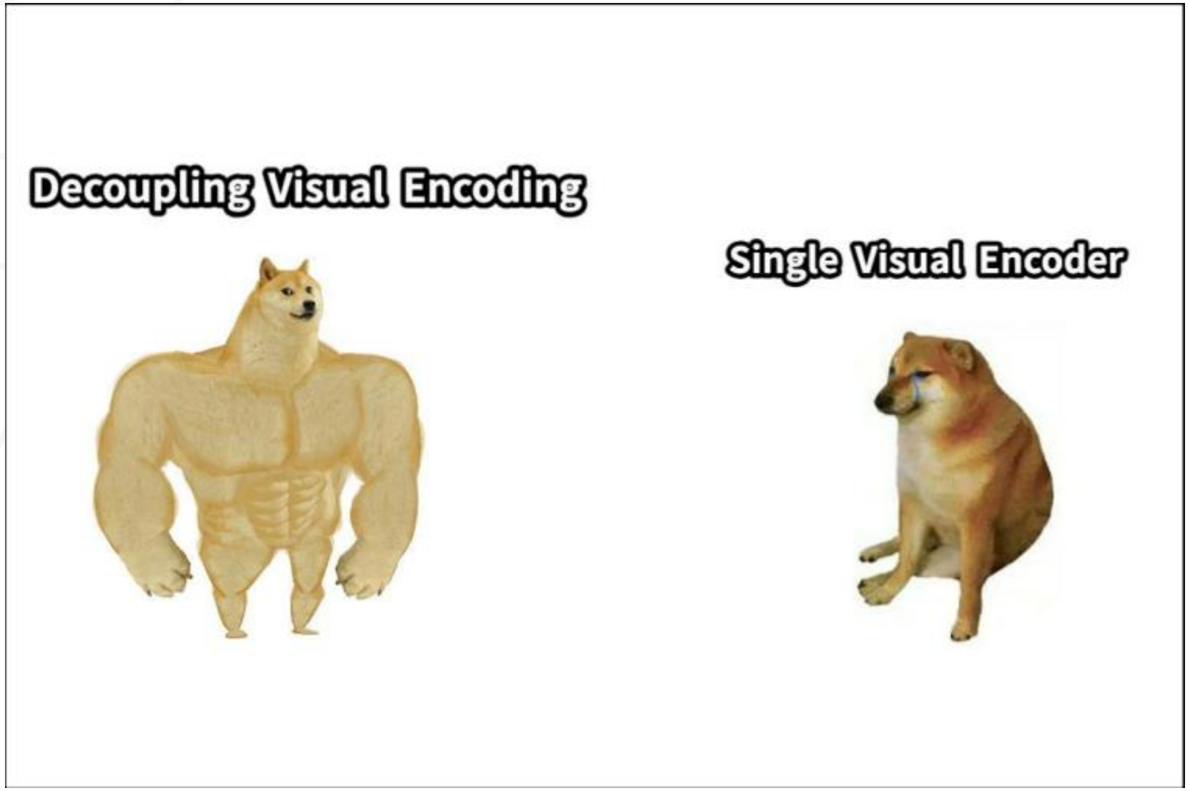


Explain this meme?

Chameleon: This image shows a cartoon of a pig and a bear side-by-side, with the text "When you're a vegan and your friends still order sausage and bacon" above them. The image is a humorous take on the challenges of being a vegan in a non-vegan environment, where one's friends or family may not share their dietary preferences.

Show-o: The image features two dogs, one brown and one white, standing side by side. The brown dog is positioned on the left side of the image, while the white dog is on the right. Both dogs are looking at the camera, seemingly posing for a picture. The brown dog is slightly larger than the white dog. The image is captioned with the phrase "I'm not a dog, but I'm a dog." This humorous statement is accompanied by the image of the two dogs, emphasizing their ...

Results



Explain this meme?

Janus (Ours): The meme is comparing two different types of neural networks used in machine learning. The first type is called "Decoupling Visual Encoding," and the second type is called "Single Visual Encoder." The meme is trying to convey that the first type is more powerful and can handle more complex tasks, while the second type is simpler and less powerful.

Georgia Tech

Generated Images

a cute fluffy chubby marmot sunbathing on a pile of rocks, snow mountains background, turquoise glacier lake afar, clear blue sky, highly detailed, golden hour, natural light, octane render, unreal engine

epic 3d portrait of white King Kong wearing mech armor made of black crystals, golden ornate around the armor, symmetrical body, hyperrealistic, intricate details, shiny, cinematic, unreal engine, artstation, octane render.

Tiny cute adorable mouse dressed as a king in a castle, anthropomorphic, Jean-Baptiste Monge, soft cinematic lighting, 8k, intricate details, portrait, Pixar style character, old fashioned movie style

Upscaled into 1024 × 1024

A close-up high-contrast photo of Sydney Opera House sitting next to Eiffel tower, under a blue night sky of roiling energy, exploding yellow stars, and radiating swirls of blue.

A wise old owl with golden plumage perched on a luminous crystal tree in a magical forest. Radiant fireflies swirl around while ethereal mist rolls through the trees, illuminated by swirls of iridescent moonlight and glistening emerald leaves.

A detailed portrait of the Roman god Janus, featuring his two faces looking in opposite directions. One face appears aged, with deep-set wrinkles and a wise, contemplative expression, while the other face is youthful, exuding vigor and curiosity. His hair is styled in flowing curls, framing both faces with a sense of divine symmetry. The artwork is rich in contrasting colors, with the left side dominated by cold blues and silvers, symbolizing winter and reflection, and the right side awash with warm golds and reds, representing spring and renewal. The background is a celestial tapestry, adorned with stars and symbolic motifs of time and passage.

A brave dog wearing a futuristic space suit, exploring an alien planet amidst swirling dunes of stardust and meteor showers. The landscape is dotted with glowing crystal formations and ethereal terraforms, creating a surreal environment in which swirling vortexes in the sky depict the endless dance of distant galaxies.

A realistic photo of snowy mountains with sunlit peaks glistening under a clear blue sky

这是一张雪山写实照片，阳光照耀下的山峰在晴朗的蓝天下熠熠生辉。前景中高大的常青树被白雪覆盖，为这令人惊叹的高山景色增添了深度。

Une photo réaliste de montagnes enneigées dont les sommets éclairés par le soleil scintillent sous un ciel bleu limpide. Au premier plan, de grands arbres à feuilles persistantes saupoudrés de neige ajoutent de la profondeur à cette scène alpine à couper le souffle.

澄み切った青空の下、太陽に照らされた山頂が輝く雪山のリアルな写真。前景の雪に覆われた背の高い常緑樹が、息をのむようなアルプスの風景に深みを与えている。

A realistic photo of ❄️ ⛰️ with ☀️ under a clear blue sky. Tall, snow-dusted 🌲 in the foreground add depth to the breathtaking alpine scene.

Multimodal Understanding Results

Prompt

Generate the latex code of this formula.

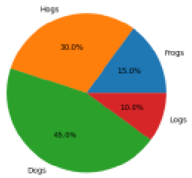
$$A_n = a_0 \left[1 + \frac{3}{4} \sum_{k=1}^{n} \left(\frac{4}{9} \right)^k \right]$$

Response

A _ { n } = a_ 0 \left[1 + \frac{3}{4} \sum_{ k=1}^{n} \left(\frac{4}{9} \right)^k \right]

Prompt

Generate the matplotlib.pyplot code for this plot.



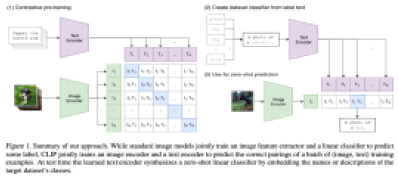
Response

The image you've provided is a pie chart that represents the distribution of different types of animals based on their frequency. The chart is divided into four segments, each representing a different type of animal and its corresponding percentage.
Here's the code to generate the pie chart:

```python
import matplotlib.pyplot as plt
# Data
animals = [
    {"name": "Hogs", "frequency": 30.0},
    {"name": "Frog", "frequency": 15.0},
    {"name": "Logs", "frequency": 10.0},
    {"name": "Dogs", "frequency": 45.0}
]
# Create the pie chart
plt.pie(animals, labels=animals, autopct='%1.1f%%',
startangle=90)
# Set the title and labels
plt.title('Distribution of Animals')
plt.title.set_text('Frequency of Animals')
# Show the plot
plt.show()
```

Prompt

What algorithm does the image depict?

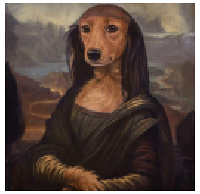


Response

The image depicts a CLIP (Contrastive Language-Image Pre-training) model. The CLIP model is a type of pre-training algorithm that combines the power of language and vision to learn a unified representation of the world. The image encoder processes the input image, and the text encoder processes the text associated with the image. The output of the text encoder is then used to predict the correct pair of images, which is a key component of the CLIP model.

Prompt

Why is this painting interesting?



Response

This image is an interesting piece of art because it combines a well-known painting, the "Mona Lisa," with a dog. The dog is depicted in the same pose as the Mona Lisa, but with a different expression and a different background. The painting is a unique blend of art styles and subjects, making it stand out.

Ablation Study

Table 5 | **Ablation studies**. We verify the effectiveness of decoupling visual encoding and compare unified training with task-specific training. "Und.", "Gen." and "SE. Tokenizer" denote "understanding", "generation" and "semantic tokenizer", respectively.

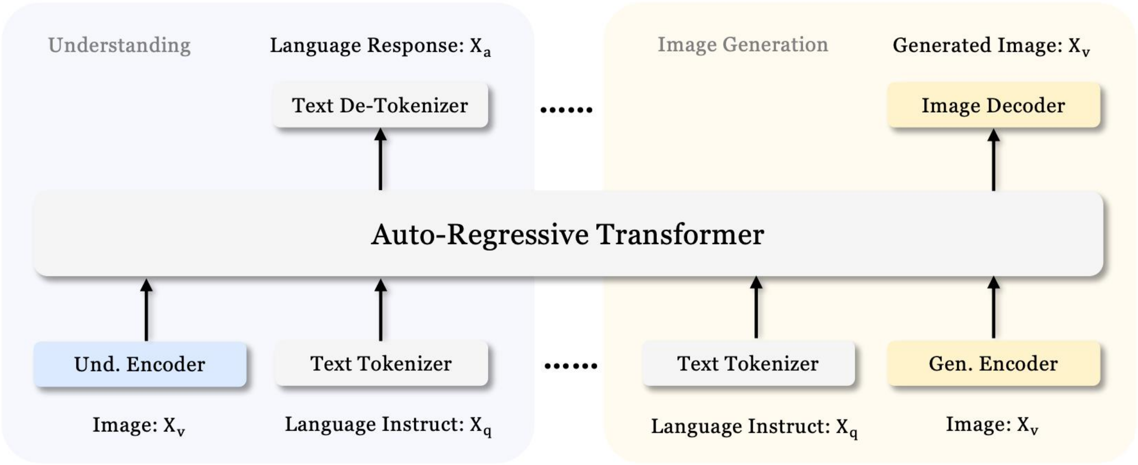
Exp ID	Visual Encoder	Training Task	POPE↑	MMB↑	SEED↑	MMMU↑	COCO-FID↓
A	VQ Tokenizer	Und. + Gen.	60.1	35.0	34.9	24.7	8.72
B	SE. Tokenizer	Und. + Gen.	82.4	52.7	54.9	26.6	7.11
C	SE. Tokenizer	Und.	83.9	62.1	60.8	27.5	-
D	SigLIP + VQ (Ours)	Und. + Gen.	87.0	69.4	63.7	30.5	8.53
E	SigLIP	Und.	85.9	70.6	64.8	28.8	-
F	VQ Tokenizer	Gen.	-	-	-	-	8.92

Key Questions
- Does decoupling work?
- How important is each component?

- **Shared encoder hurts understanding** → Exp-A weak scores.
- **Semantic encoder improves semantics** but still shows trade-offs (Exp-B vs Exp-C).
- **Decoupling (Exp-D)** → strong understanding **and** competitive generation.
- **Unified training (Exp-D)** ≈ Task-specific (Exp-E/F) → efficiency without loss.

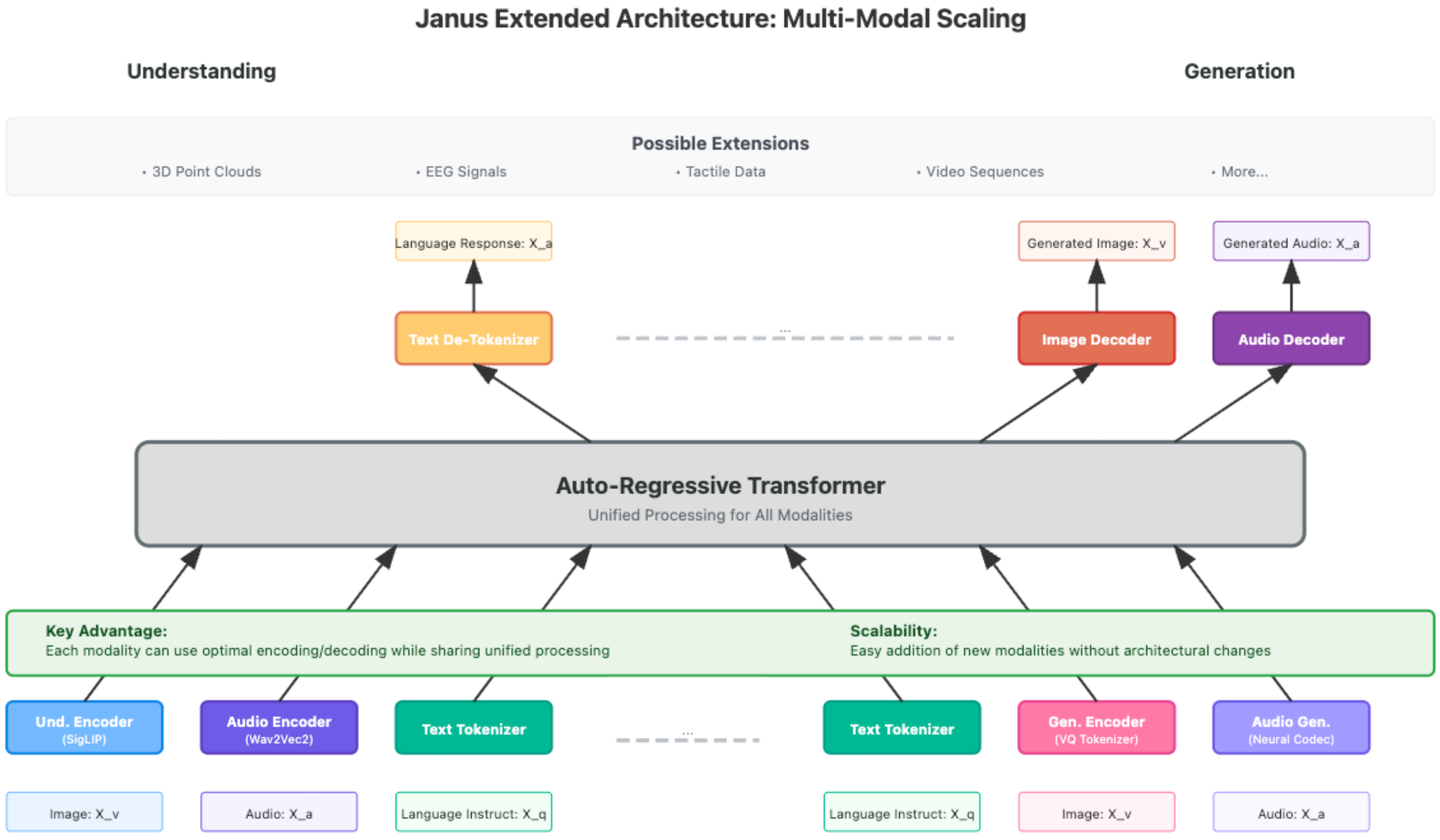
Scaling the Model

Can we add other modalities?



Strengths

Why Not!



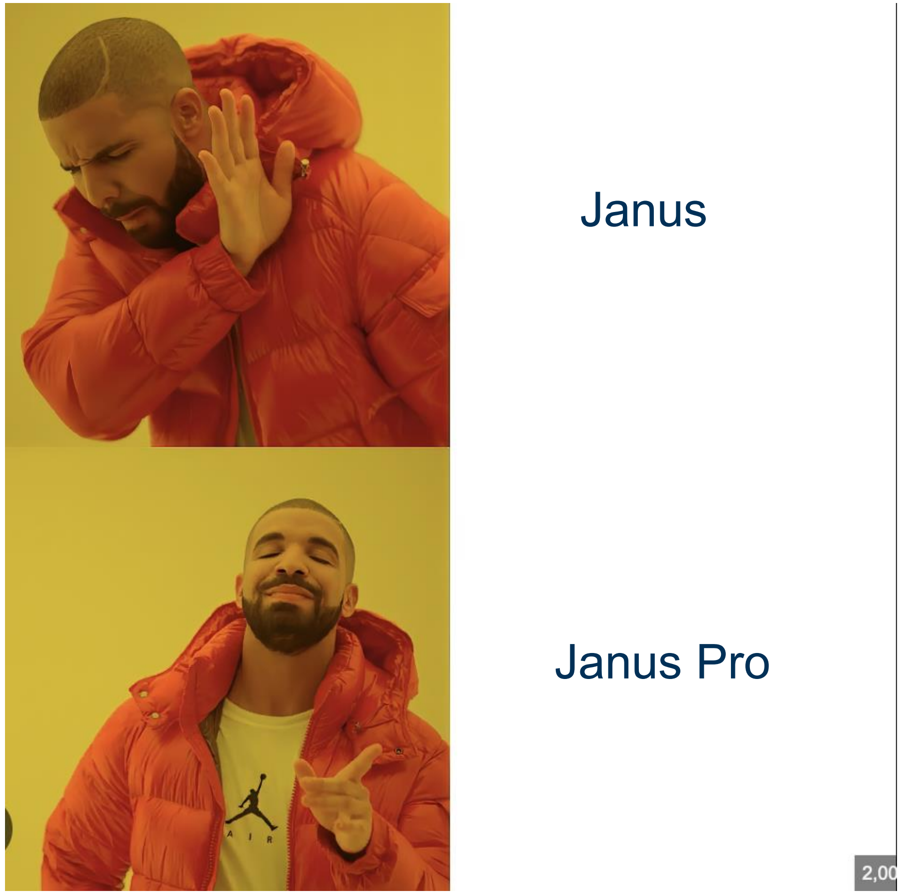
- Flexibility & Extensibility
- Outperforms previous unified models while matching larger task-specific models with fewer parameters
- First to identify and solve the fundamental conflict between understanding and generation encoding needs

Janus-Pro: Unified Multimodal Understanding and Generation with Data and Model Scaling

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan
DeepSeek-AI
29th Jan 2025

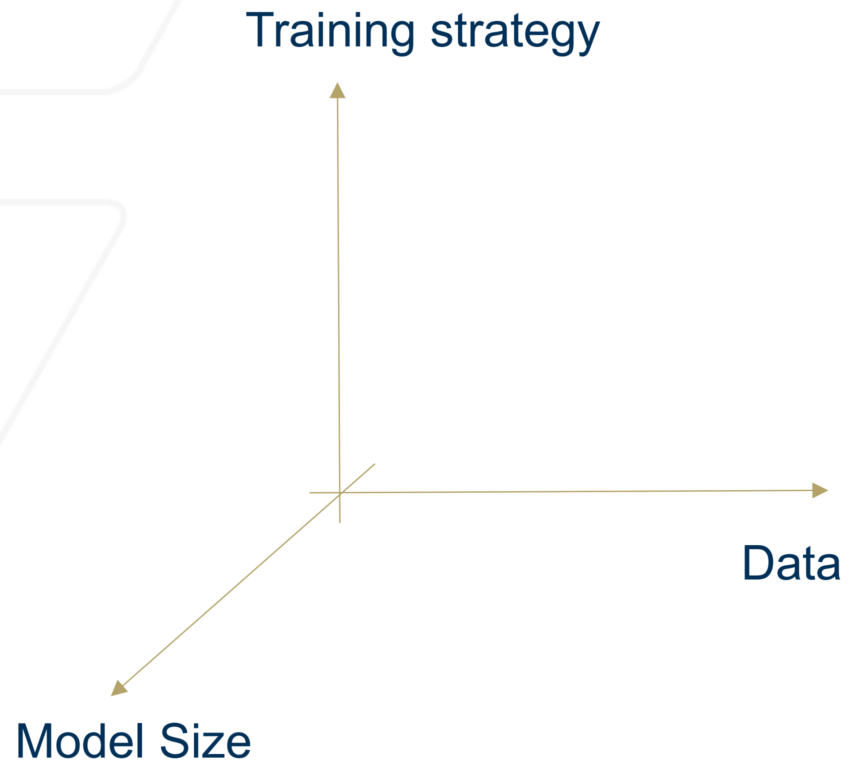
Janus to Janus Pro

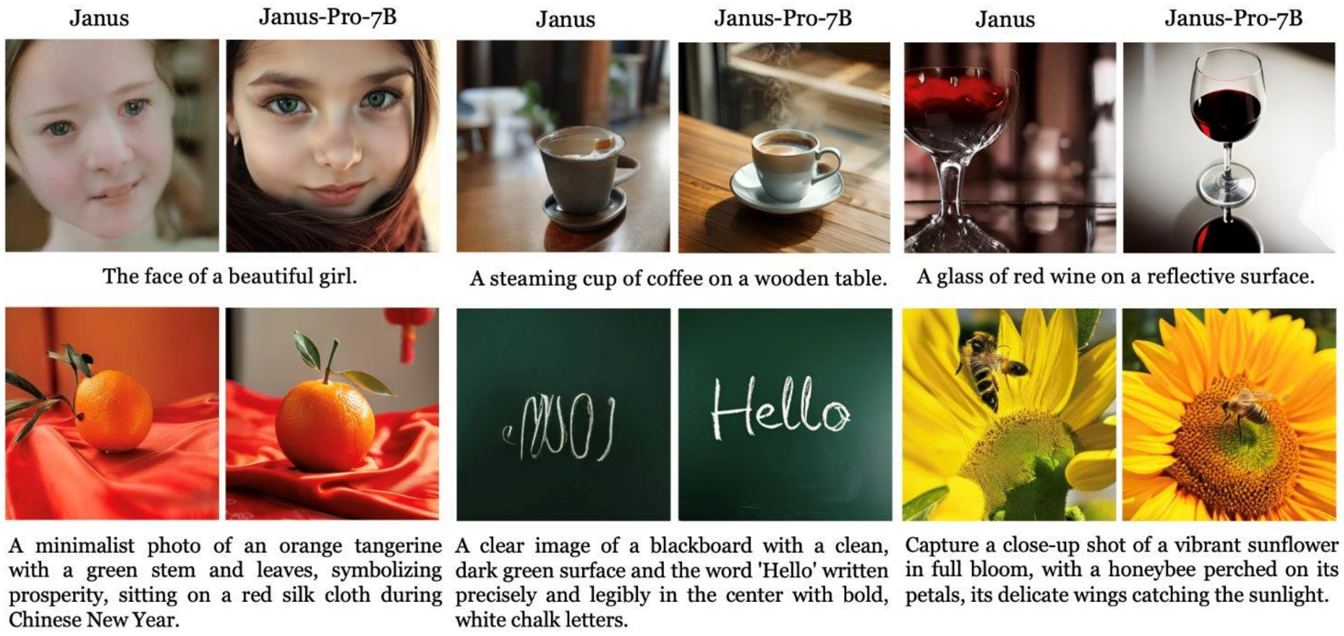
- 1B parameters scale
- Limited training data
- Limited model capacity
- Subpar at short prompts image generation
- Quality of images



Georgia Tech

Features of Janus Pro





Axis 1: Training Strategy

- Issues with Stage 2 Training from Janus
  - Part 1: ImageNet category names as prompts
  - Part 2: normal text to image data
  - 66.6% of training steps allocated to part 1
- Modifications:
  - Longer training in stage 1 on ImageNet
  - Drop ImageNet data in stage 2
- Enables Stage 2 to utilize the text-to-image data more efficiently
- Adjust the data ratio in stage 3. ***Why?***

Multimodal	Pure Text	Text-to-Image
7	3	10
5	1	4 ⇩

Ratios for data in stage 3

Empirical: maintain strong visual generation capabilities while achieving improved multimodal understanding performance

Axis 2: Data

- Multimodal data
  - Stage 2: + 90 M samples, "like": YFCC, Docmatix
  - Stage 3: + MEME understanding, Chinese conversational data, "datasets aimed at enhancing dialogue experiences"
- Visual Generation
  - real world data noisy => 72 million samples of synthetic aesthetic data. 1:1 for synthetic and real

- ***Why does the model converge on synthetic data faster?***

Synthetic Data Samples from Midjourney used by Janus Pro training

cats with many eyes floating in colorful glowing swirling whisps, occult inspired, emerging from the void, shallow depth of field

Axis 3: Model Scale

- From 1.5B to 7B
- when utilizing a larger-scale LLM, the convergence speed of losses for both multimodal understanding and visual generation improved significantly compared to the smaller model.
    - ***Why?***

Table 1 | **Architectural configuration for Janus-Pro**. We list the hyperparameters of the architecture.

	Janus-Pro-1B	Janus-Pro-7B
Vocabulary size	100K	100K
Embedding size	2048	4096
Context Window	4096	4096
#Attention heads	16	32
#Layers	24	30

Georgia Tech

Multimodal Understanding

Table 3 | **Comparison with state-of-the-arts on multimodal understanding benchmarks**. "Und." and "Gen." denote "understanding" and "generation", respectively. Models using external pretrained diffusion model are marked with †.

Type	Model	# LLM Params	POPE↑	MME-P↑	MMB↑	SEED↑	GQA↑	MMMU↑	MM-Vet↑
Und. Only	LLaVA-v1.5-Phi-1.5 [50]	1.3B	84.1	1128.0	-	-	56.5	30.7	-
	MobileVLM [6]	1.4B	84.5	1196.2	53.2	-	56.1	-	-
	MobileVLM-V2 [7]	1.4B	84.3	1302.8	57.7	-	59.3	-	-
	MobileVLM [6]	2.7B	84.9	1288.9	59.6	-	59.0	-	-
	MobileVLM-V2 [7]	2.7B	84.7	1440.5	63.2	-	61.1	-	-
	LLaVA-Phi [56]	2.7B	85.0	1335.1	59.8	-	-	-	28.9
	LLaVA [27]	7B	76.3	809.6	38.7	33.5	-	-	25.5
	LLaVA-v1.5 [26]	7B	85.9	1510.7	64.3	58.6	62.0	35.4	31.1
	InstructBLIP [8]	7B	-	-	36.0	53.4	49.2	-	26.2
	Qwen-VL-Chat [1]	7B	-	1487.5	60.6	58.2	57.5	-	-
	IDEFICS-9B [19]	8B	-	-	48.2	-	38.4	-	-
	Emu3-Chat [45]	8B	85.2	1244	58.5	68.2	60.3	31.6	37.2
	InstructBLIP [8]	13B	78.9	1212.8	-	-	49.5	-	25.6
Und. and Gen.	DreamLLM† [10]	7B	-	-	-	-	-	-	36.6
	LaVIT† [18]	7B	-	-	-	-	46.8	-	-
	MetaMorph† [42]	8B	-	-	75.2	71.8	-	-	-
	Emu† [39]	13B	-	-	-	-	-	-	-
	NExT-GPT† [47]	13B	-	-	-	-	-	-	-
	Show-o-256 [50]	1.3B	73.8	948.4	-	-	48.7	25.1	-
	Show-o-512 [50]	1.3B	80.0	1097.2	-	-	58.0	26.7	-
	D-Dit [24]	2.0B	84.0	1124.7	-	-	59.2	-	-
	Gemini-Nano-1 [41]	1.8B	-	-	-	-	-	26.3	-
	ILLUME [44]	7B	88.5	1445.3	65.1	72.9	–	38.2	37.0
	TokenFlow-XL [34]	13B	86.8	1545.9	68.9	68.7	62.7	38.7	40.7
	LWM [28]	7B	75.2	-	-	-	44.8	-	9.6
	VILA-U [48]	7B	85.8	1401.8	-	59.0	60.8	-	33.5
	Chameleon [40]	7B	-	-	-	-	-	22.4	8.3
	Janus	1.5B	87.0	1338.0	69.4	63.7	59.1	30.5	34.3
	Janus-Pro-1B	1.5B	86.2	1444.0	75.5	68.3	59.3	36.3	39.8
	Janus-Pro-7B	7B	87.4	1567.1	79.2	72.1	62.0	41.0	50.0

Georgia Tech

Visual Generation

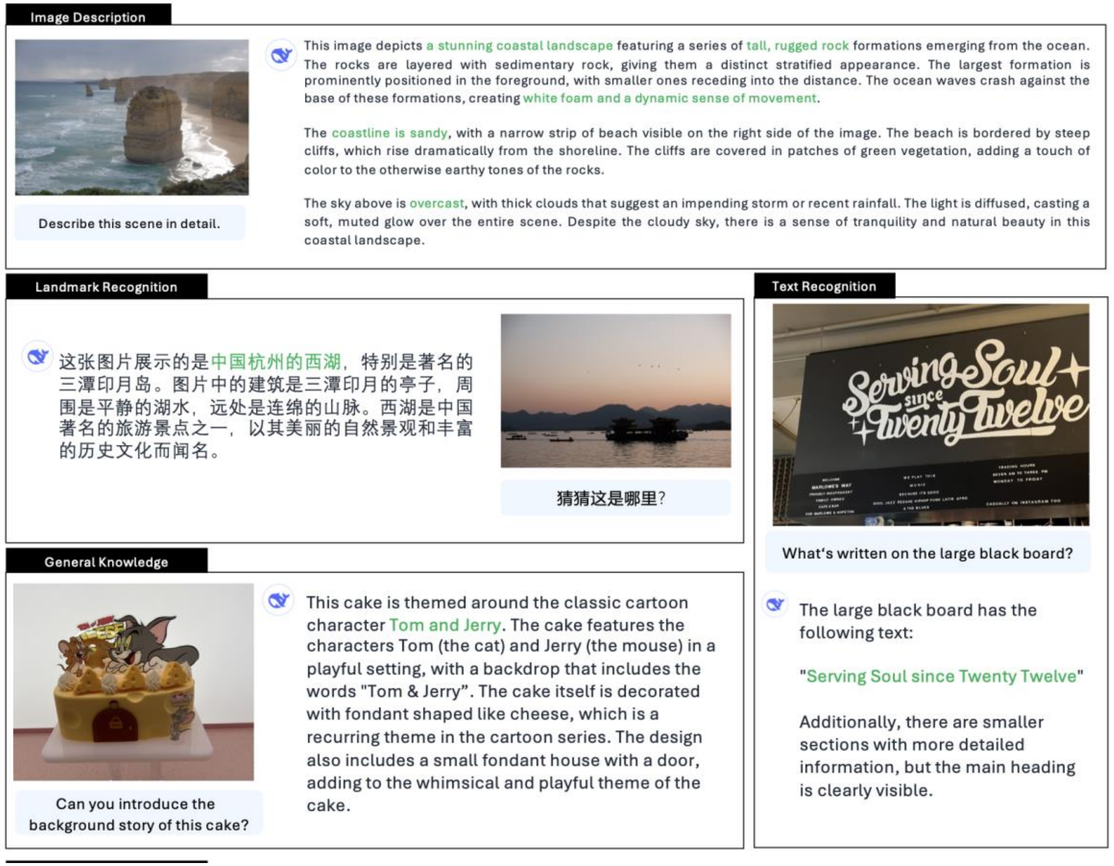
Table 5 | **Performances on DPG-Bench.** The methods in this table are all generation-specific models except Janus and Janus-Pro.

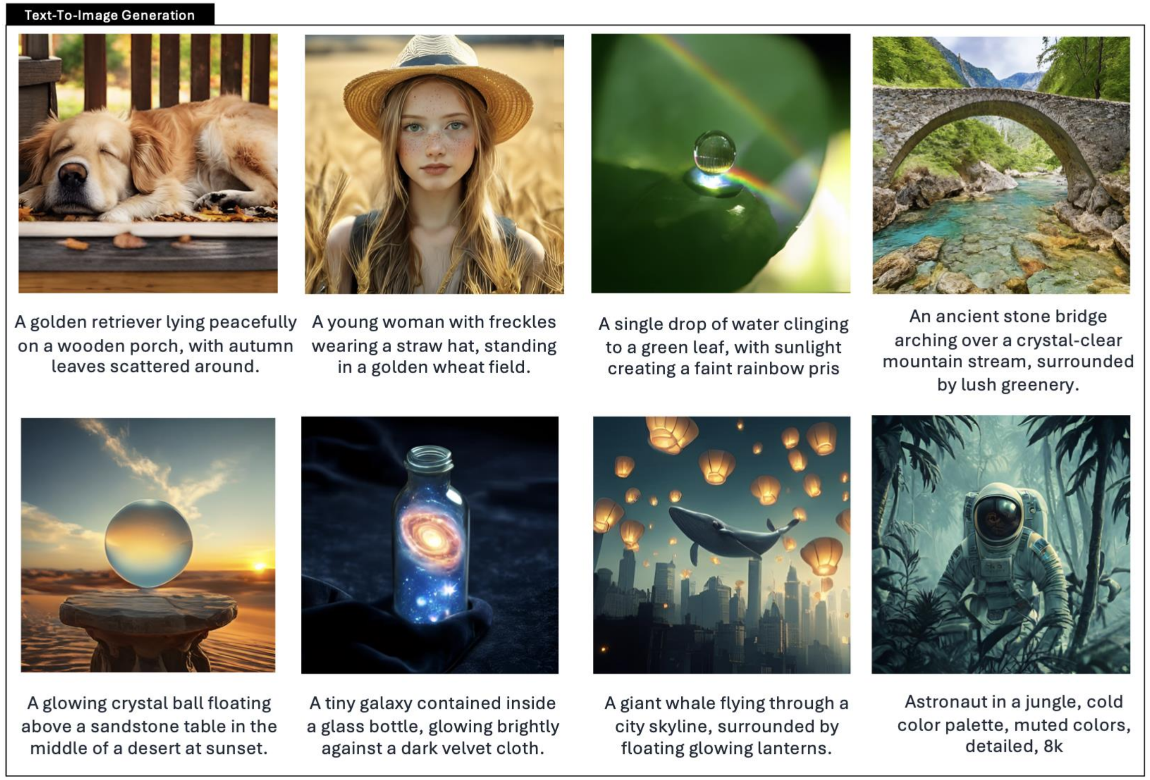
Method	Global	Entity	Attribute	Relation	Other	Overall↑
SDv1.5 [36]	74.63	74.23	75.39	73.49	67.81	63.18
PixArt-α [4]	74.97	79.32	78.60	82.57	76.96	71.11
Lumina-Next [57]	82.82	88.65	86.44	80.53	81.82	74.63
SDXL [33]	83.27	82.43	80.91	86.76	80.41	74.65
Playground v2.5 [22]	83.06	82.59	81.20	84.08	83.50	75.47
Hunyuan-DiT [25]	84.59	80.59	88.01	74.36	86.41	78.87
PixArt-Σ [5]	86.89	82.89	88.94	86.59	87.68	80.54
Emu3-Gen [45]	85.21	86.68	86.84	90.22	83.15	80.60
DALL-E 3 [2]	90.97	89.61	88.39	90.58	89.83	83.50
SD3-Medium [11]	87.90	91.01	88.83	80.70	88.68	84.08
Janus	82.33	87.38	87.70	85.46	86.41	79.68
Janus-Pro-1B	87.58	88.63	88.17	88.98	88.30	82.63
Janus-Pro-7B	86.90	88.90	89.40	89.32	89.48	84.19

Table 4 | **Evaluation of text-to-image generation ability on GenEval benchmark.** "Und." and "Gen." denote "understanding" and "generation", respectively. Models using external pretrained diffusion model are marked with †.

Type	Method	Single Obj.	Two Obj.	Counting	Colors	Position	Color Attri.	Overall↑
Gen. Only	LlamaGen [38]	0.71	0.34	0.21	0.58	0.07	0.04	0.32
	LDM [37]	0.92	0.29	0.23	0.70	0.02	0.05	0.37
	SDv1.5 [37]	0.97	0.38	0.35	0.76	0.04	0.06	0.43
	PixArt-α [4]	0.98	0.50	0.44	0.80	0.08	0.07	0.48
	SDv2.1 [37]	0.98	0.51	0.44	0.85	0.07	0.17	0.50
	DALL-E 2 [35]	0.94	0.66	0.49	0.77	0.10	0.19	0.52
	Emu3-Gen [45]	0.98	0.71	0.34	0.81	0.17	0.21	0.54
	SDXL [32]	0.98	0.74	0.39	0.85	0.15	0.23	0.55
	DALL-E 3 [2]	0.96	0.87	0.47	0.83	0.43	0.45	0.67
	SD3-Medium [11]	0.99	0.94	0.72	0.89	0.33	0.60	0.74
Und. and Gen.	SEED-X† [13]	0.97	0.58	0.26	0.80	0.19	0.14	0.49
	Show-o [50]	0.95	0.52	0.49	0.82	0.11	0.28	0.53
	D-DiT [24]	0.97	0.80	0.54	0.76	0.32	0.50	0.65
	LWM [28]	0.93	0.41	0.46	0.79	0.09	0.15	0.47
	Transfusion [55]	-	-	-	-	-	-	0.63
	ILLUME [44]	0.99	0.86	0.45	0.71	0.39	0.28	0.61
	TokenFlow-XL [28]	0.95	0.60	0.41	0.81	0.16	0.24	0.55
	Chameleon [40]	-	-	-	-	-	-	0.39
	Janus [46]	0.97	0.68	0.30	0.84	0.46	0.42	0.61
	Janus-Pro-1B	0.98	0.82	0.51	0.89	0.65	0.56	0.73
	Janus-Pro-7B	0.99	0.89	0.59	0.90	0.79	0.66	0.80

Qualitative Results



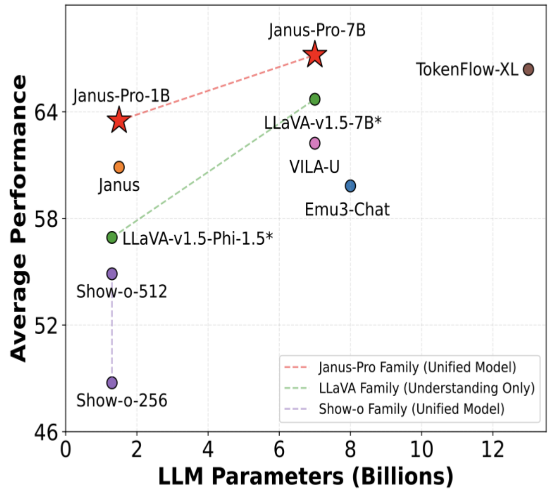


Guess the resolution? 384 ×384

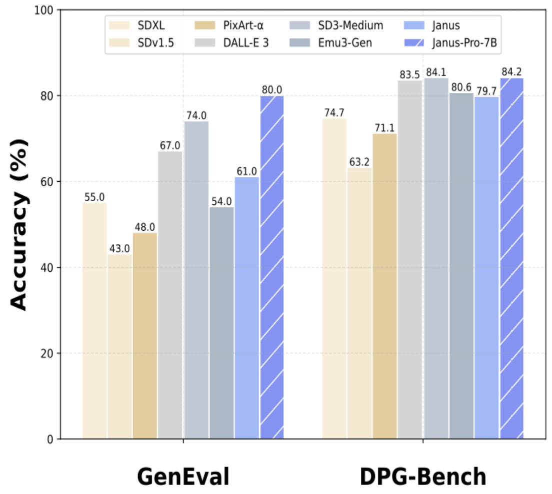
Summary of Strengths, Weaknesses, Relationships

- Improvement across 3 axes, training, data and scale
- Code and models are publicly available

- Input resolution limited to 384x384
  - Difficulties on fine-grained tasks such as OCR
  - Images lack fine detail



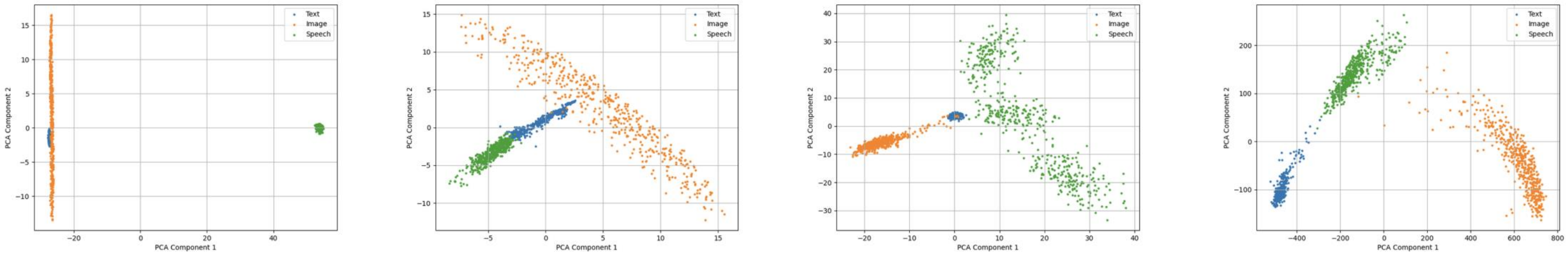
(a) Average performance on four multimodal understanding benchmarks.



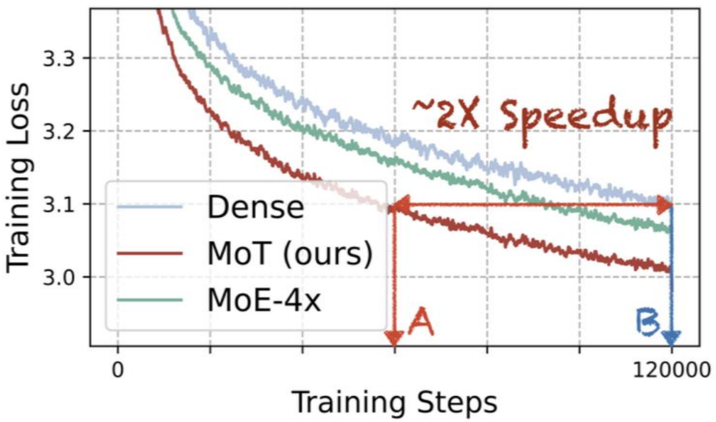
(b) Performance on instruction-following benchmarks for text-to-image generation.

Weakness of existing multi-modal early fusion modal

A note-able criticism of Chameleon Addressed by a Follow up work Mixture of Multimodal Transformers is unmixed representations



(b) Layer 1 (c) Layer 5 (d) Layer 17 (e) Layer 32



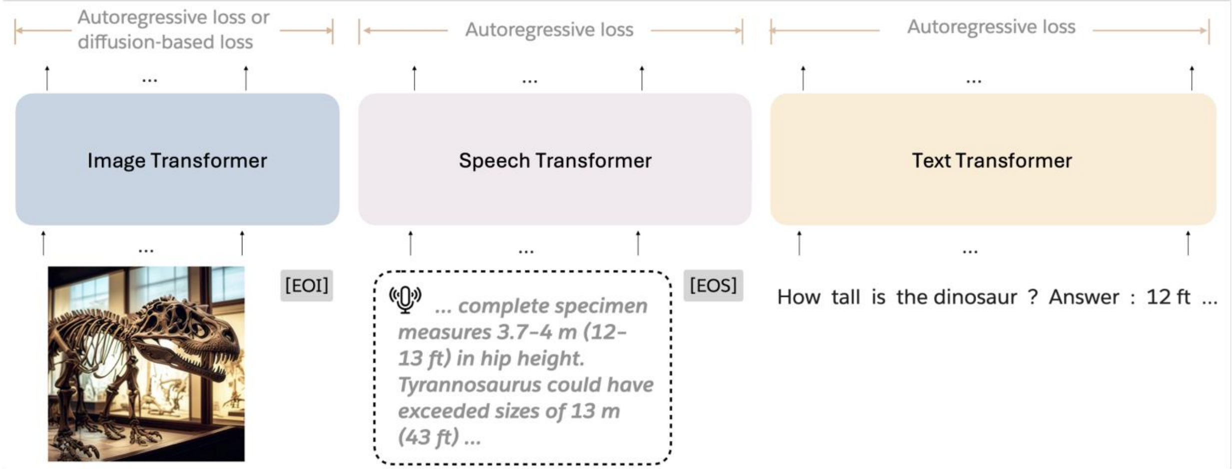


Figure 1: **Mixture-of-transformer (MoT)** for native multi-modal generative modeling.

https://arxiv.org/pdf/2411.04996

Thank You