Chameleon + BAGEL

Chameleon Team (META FAIR) - arXiv, May 2024
Deng et al. (Bytedance) - arXiv, May 2025

Shadi Buchanan, Pranav Somu, Oytun Kuday Duran

Unified multimodal understanding + generation

Outline

One sequence of questions for each paper

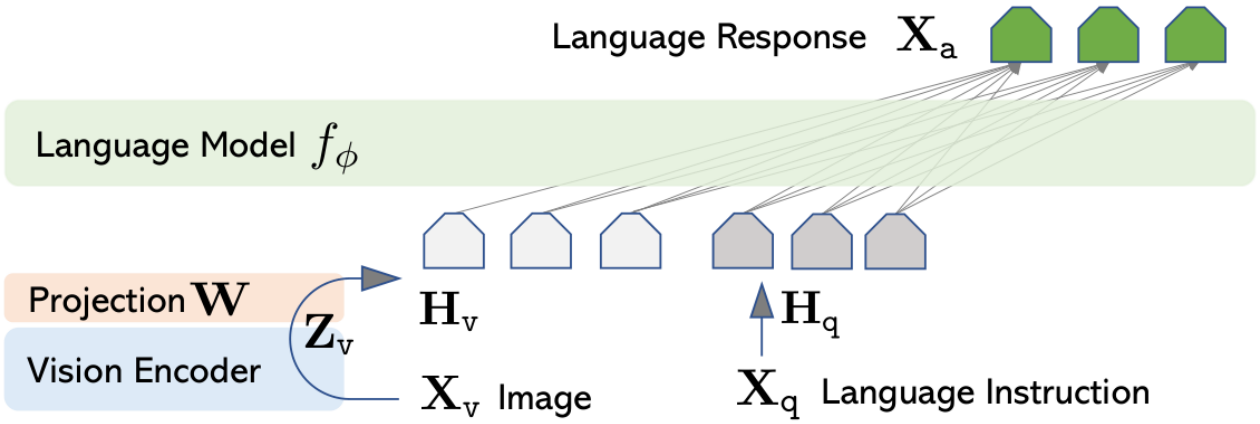
Chameleon (2024), then BAGEL (2025)

- Problem Statement
- Related Works
- Approach and Training Data
- Experiments & Results
- Limitations and Implications

Georgia Tech

Problem Statement: Chameleon

Why do image understanding and image generation need a shared context?

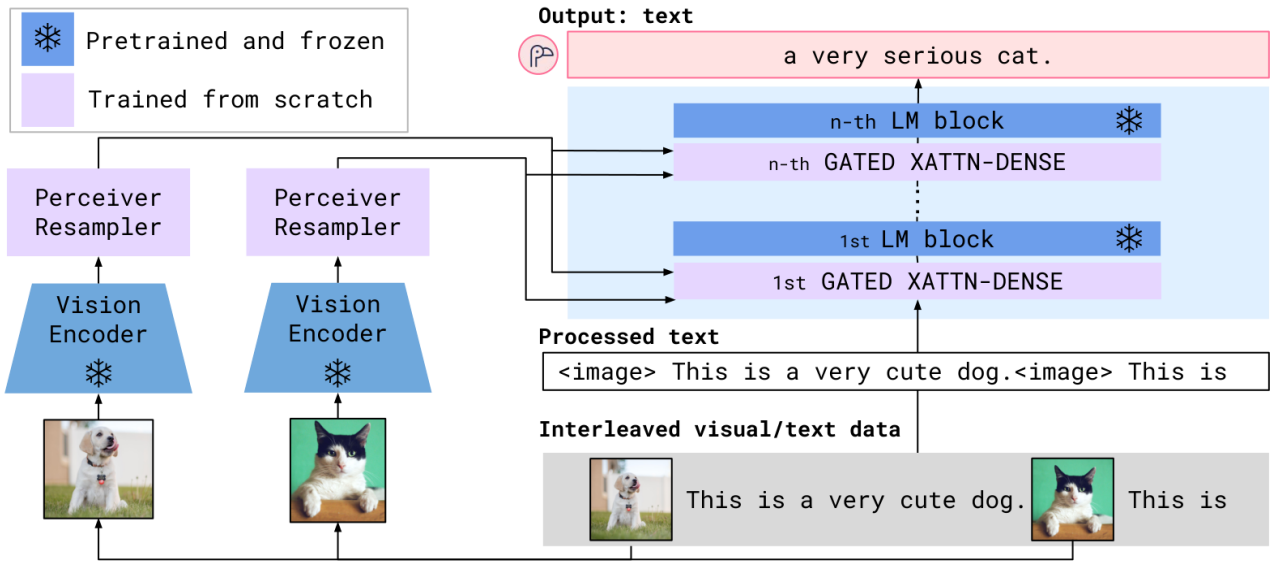


LLaVA (2023): image-conditioned language generation

- **What earlier systems did:** A vision encoder supplied image features to a language model through a learned interface.
- **What remained separate:** The model could explain an image, but its output vocabulary was language. Creating an image needed another generator.
- **Why change it:** An illustrated answer must keep the pictures, words, and previous context consistent as the response grows.
- **Chameleon's question:** Can one transformer directly continue that shared sequence with either text or images?

Related Works: Chameleon

Earlier systems made different choices about where to combine modalities



Flamingo (2022): visual cross-attention inside a language model

- **Connected vision-language models:** Flamingo and LLaVA add visual information to a language backbone. Their main output is text.
- **Discrete image generators:** DALL-E and CM3Leon predict image codebook tokens.
- **CM3: causal masked multimodal modeling:** A web document is one text/image sequence. The model reconstructs missing content using its context.
- **Chameleon's contribution:** Scale native mixed output to 7B and 34B, with stable training and instruction alignment.

Overview: Chameleon

A foundational model for multimodal autoregressive generation

Prompt: My mom just adopted a dog called a mountain cur. I thought I knew a lot about dogs but I have never heard of this breed. Can you give me some basic information as well as show me a few pictures of what they look like?

Response: Sure! Mountain Cur dogs are a type of hunting dog that originated in the Appalachian Mountains. They are known for their intelligence, loyalty, and strong prey drive. They are typically medium-sized dogs, with a muscular build and a short, easy-to-maintain coat that comes in a variety of colors including black, brown, tan, red, and white. Here is an image of a black and white mountain cur dog: **<img>**

Input

Text, images, or an interleaved sequence

Autoregressive generation

Predict the next text or discrete image token.

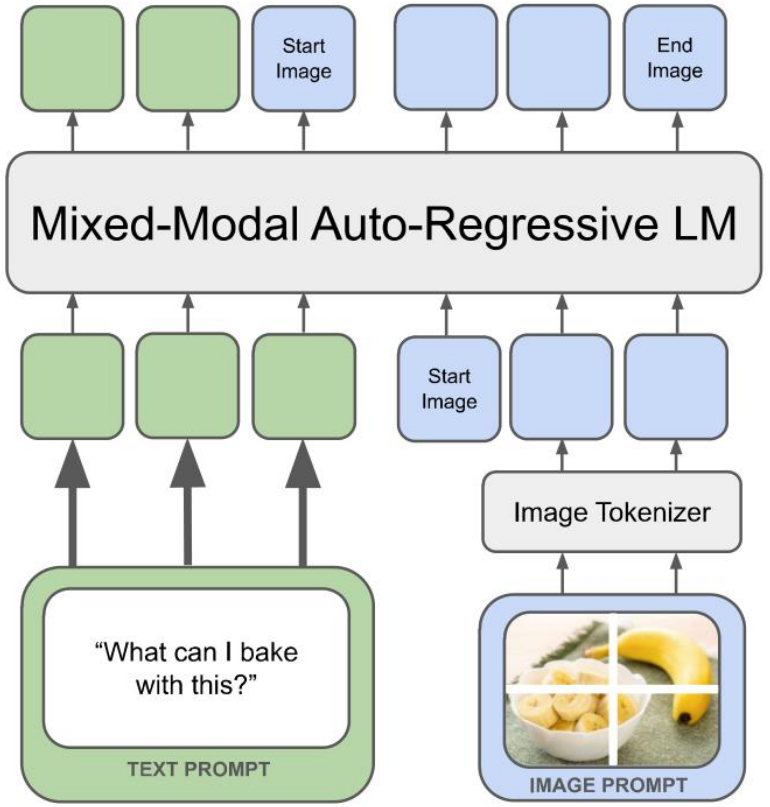
Output

Text, generated images, or both in one response

Georgia Tech

Approach: Chameleon

One mixed sequence and one dense transformer

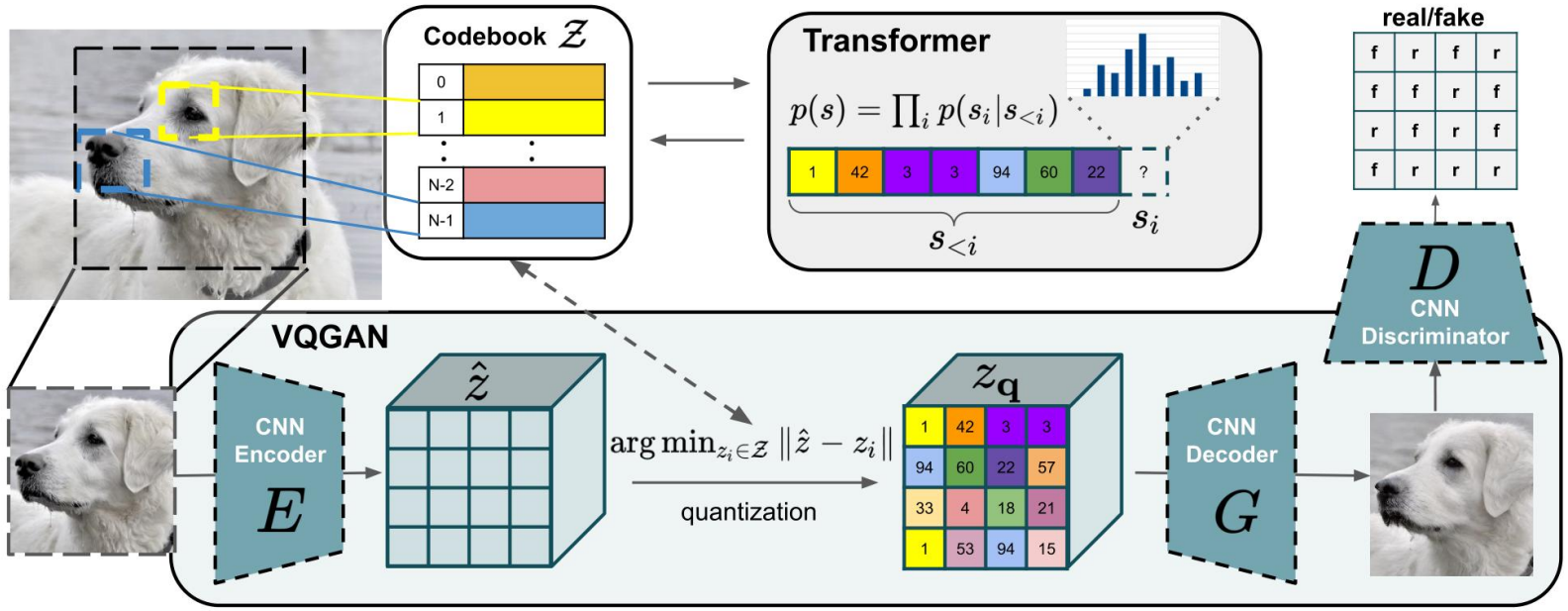


(a) Mixed-Modal Pre-Training

- **The question:** Can one model both read and generate interleaved images and text?
- **The contribution:** One token stream and one autoregressive transformer, trained from scratch
- **The scale:** 7B and 34B parameters with a recipe for stable mixed-modal training

Approach: Chameleon

Images become discrete tokens



Pixels to codes

512 × 512 pixels become a 32 × 32 grid: 1,024 image tokens.

A learned vocabulary

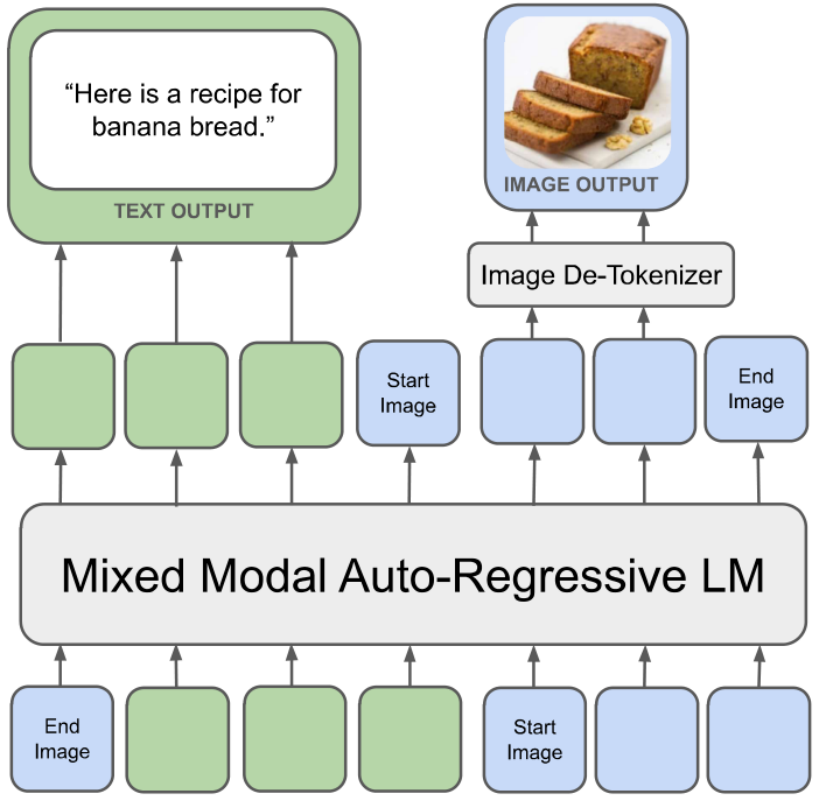
8,192 image codebook entries within a 65,536-token combined vocabulary.

Codes to pixels

The decoder reconstructs an image from the predicted discrete codes.

Approach: Chameleon

One next-token objective for text and images



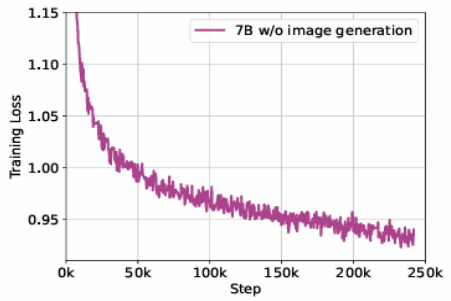
(b) Mixed-Modal Generation

- **Common objective:** Predict the next token given every preceding image and text token
- **Shared backbone:** Dense Llama-style decoder: RMSNorm, SwiGLU and rotary positions
- **Generation:** Text → image-start → 1,024 image tokens → image-end → more text

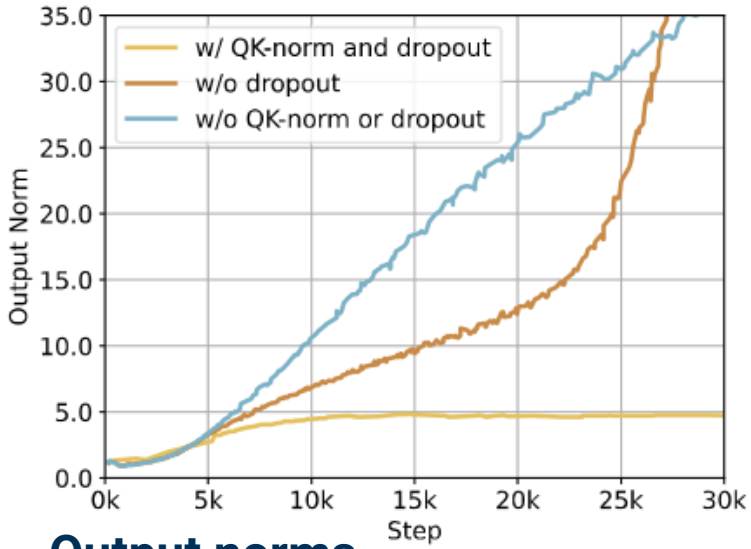
Georgia Tech

Approach: Chameleon

Why does mixed-modal training become unstable?

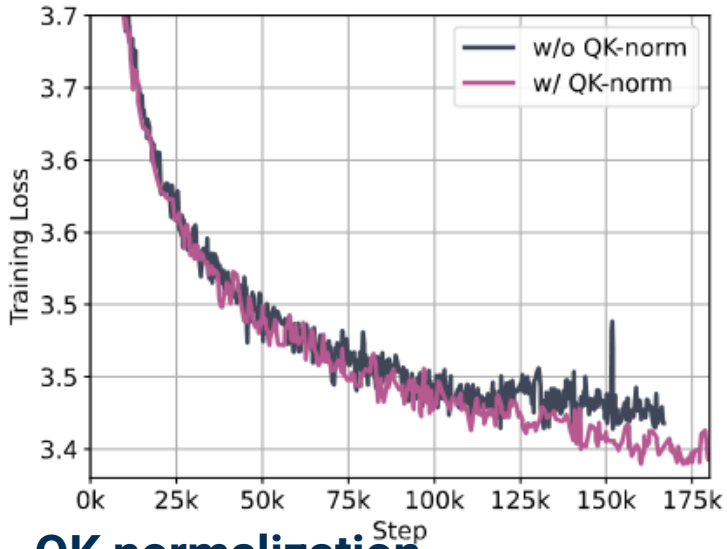


(b) Training loss curve with image generation disabled does not suffer from instability issues.



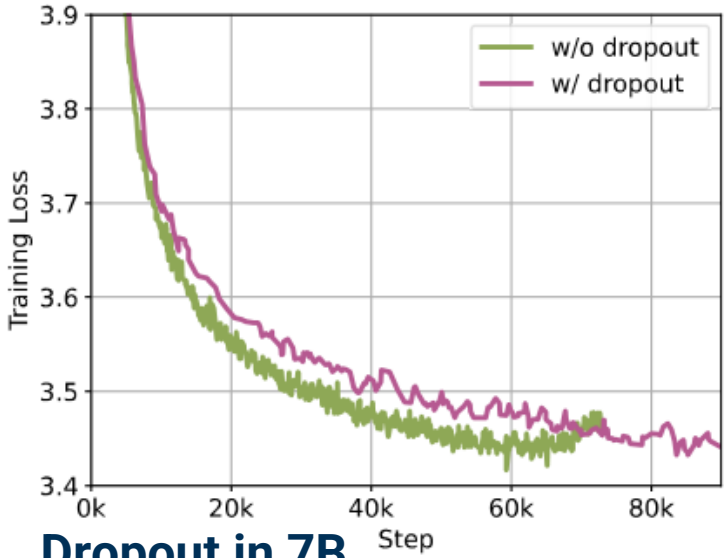
Output norms

QK normalization plus dropout restrains the growth of hidden-state norms.



QK normalization

The normalized run stays stable longer. The run without QK normalization eventually diverges.



Dropout in 7B

Dropout raises early loss slightly, but supports a longer stable run. Scaling to 34B needs another change.

Takeaway: extra instability happens when one set of weights serves two modalities' distributions. **Issue:** authors do not provide a rigorous/mathematical explanation.

Georgia Tech

Approach: Chameleon

Two softmax operations need two different controls

- **Inside attention: QK normalization:** LayerNorm on queries and keys controls the scale of attention scores before softmax.

The paper's output-logit regularizer

The application of QK-Norm while helping the inner softmaxes within the Transformer does not solve the problem of logit shift in the final softmax. Following Chowdhery et al. (2022); Wortsman et al. (2023), 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.

- **Logit shift:** $\mathrm{softmax}(\mathbf{x} + c) = \mathrm{softmax}(\mathbf{x})$ so uniform logit increase/decrease is uncontrolled.

Approach: Chameleon

Why moving normalization stabilizes the 34B residual blocks

Chameleon-34B:

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

Llama2:

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

Llama 2: normalize the input

The attention / feed-forward branch can still produce a large update before it is added back.

Chameleon-34B: normalize the update

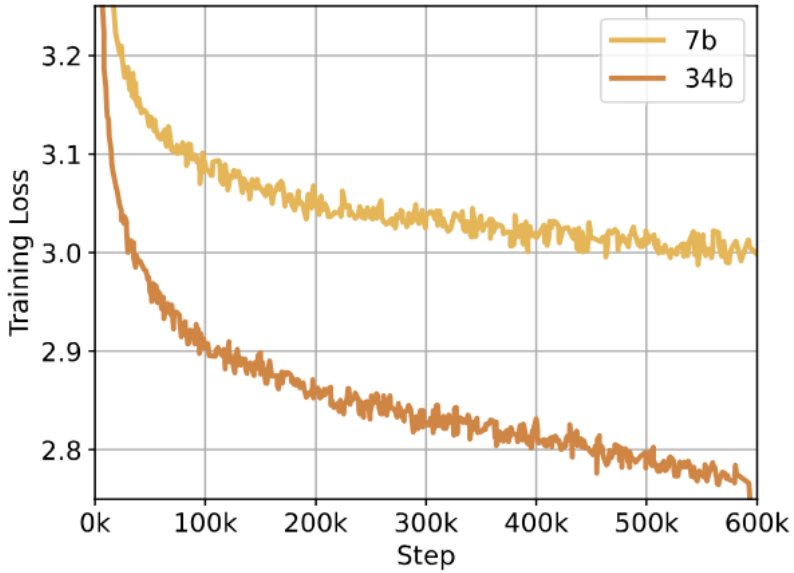
Apply RMSNorm after each branch, before the residual addition. This controls the scale of the added update.

The skip connection stays intact. Normalize each branch output, not the entire residual sum.

Data: Chameleon

How was Chameleon pretrained, and what did each data source teach?

Training data	Reported scale	What the model learns
Text and code	2.9T tokens	Language and code
Image-text pairs	1.4B pairs	Image-to-text and text-to-image
Interleaved web documents	400B tokens	Context across text and images



Chameleon pretraining loss over the first 600K steps

Takeaway: with no pretrained backbone to inherit, language has to be learned again alongside pixels: 9.2T tokens over 2.1 epochs, from scratch. We could have instead taken language ability off-the-shelf.

Training and Alignment: Chameleon

Why pretraining needs instruction fine-tuning

Prompt: Which elements contribute to this flower arrangement's distinctiveness, and what sets its surroundings apart?
Response: The image features an autumnal-themed floral arrangement in a classically designed vase. The arrangement stands out with its striking combination of red maple leaves, purple flowers, and pink-red roses. The vase's rosettes give it an antique design. In the background, blurry green vegetation is visible, creating a contrast to the autumnal theme of the bouquet. The red maple leaves, both in the arrangement and scattered around the vase, evoke autumn, while the green vegetation suggests a more summer-like setting. The outdoor setting is bright and sunny, adding to the contrast between the arrangement and its settings.

(d) Visual Chat

Gyumri is an urban municipal community and the second-largest city in Armenia, serving as the administrative center of Shirak Province in the northwestern part of the country.

Archaeological excavations conducted throughout the Soviet period have shown that the area of modern-day Gyumri has been populated since at least the third millennium BC.

(e) Interleaved Text/Image Generation

- **Why alignment:** Document continuation does not guarantee a helpful answer or the right choice of text versus images.
- **What changes:** Supervised prompt–answer examples cover text, code, visual chat, image generation, interleaved generation, and safety.
- **How it trains:** Compute loss only on answer tokens. Balance modalities to avoid always generating images or avoiding them.

Experiments & Results: Chameleon

Language benchmarks test whether multimodality sacrifices text ability

Benchmark	Chameleon 34B	Llama-2 34B	Llama-2 70B
MMLU, 5-shot	65.8	62.6	68.9
GSM8K, 8-shot, maj@1	61.4	42.2	56.8

- **Interpretation:** Text ability is retained, but extra training tokens and data type were used, so adding a modality doesn't necessarily mean improved or worse text performance (capabilities can coexist).

Prompt: Given a list of integers, find the first number that appears at least twice.

Response:

```
def find_duplicate(my_list):
    seen = set()
    for num in my_list:
        if num in seen:
            return num
        seen.add(num)
    return None
```

(b) Code

Code fine-tuning example.

Experiments & Results: Chameleon

Captioning gains coexist with a visual-question-answering gap

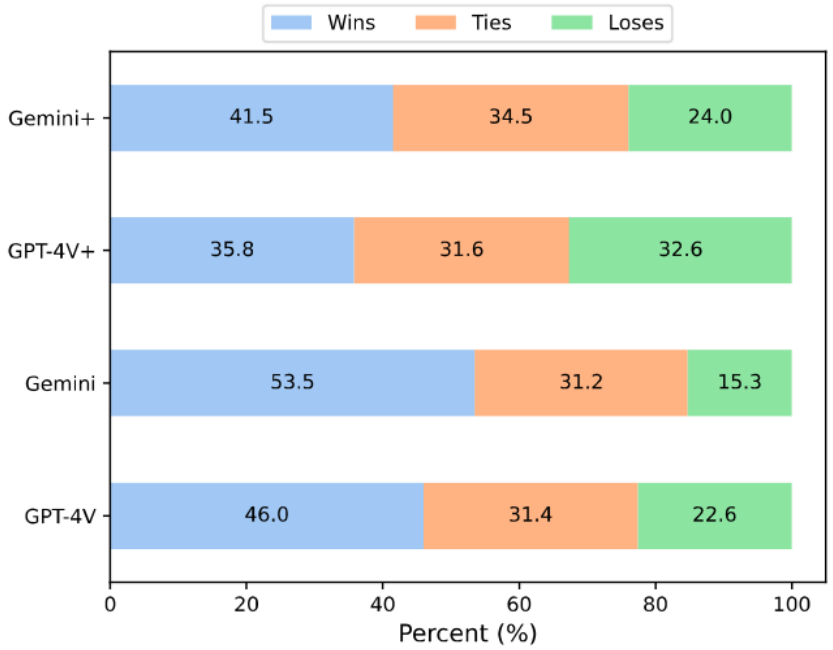
Task / metric	Chameleon-34B	Comparison / setting
COCO / CIDEr	Pretrained: 120.2 2-shot	IDEFICS-80B: 116.6 32-shot
Caption SFT / CIDEr	COCO: 140.8, 0-shot Flickr30k: 82.3, 2-shot	Flamingo-80B-FT: COCO 138.1 Gemini Pro: Flickr30k 82.2, 4-shot
VQAv2 test-dev	MultiTask: 69.6 2-shot	Flamingo-80B-FT: 82.0

Image from the paper's qualitative visual-chat example

Takeaway: early fusion buys description more than inspection. 120.2 CIDErat 2 shots beats an 80B model at 32, while VQAv2 trails: the 1,024-code bottleneck ("quantization") drops the detail questions ask about.

Experiments & Results: Chameleon

Chameleon wins, ties, and losses on mixed-output prompts



Takeaway: the win is in fulfilling a mixed request, not in image quality. Against the stronger GPT-4V+ pipeline it is 35.8% wins to 32.6% losses, close to a tie.

- **Study design:** 1,048 prompts requiring mixed output. Three raters per judgment.
- **Task fully fulfilled:** Chameleon: 55.2%
- Gemini+: 37.6%
- GPT-4V+: 44.7%
- **Meaning of "+":** The API model writes captions, then DALL-E 3 generates the images.

Limitations & Implications: Chameleon

Native multimodal documents are feasible, but full sharing has costs

Prompt: <img> My plant looks like this... Is this a problem? If so, how do I fix it?

Response: The image you provided shows a pothos plant with yellowing leaves, which can be caused by a variety of factors, including overwatering, underwatering, lack of nutrients, too much direct sunlight, or pests. Here are some steps you can take to troubleshoot and potentially fix the issue:

- Check the soil: Pothos plants prefer well-draining soil that is neither too wet nor too dry. If the soil feels soggy or waterlogged, stop watering for a while. Conversely, if the soil feels dry to the touch, it's time to water.
- Adjust watering: Overwatering is a common cause of yellowing leaves on pothos plants. Reduce the frequency of watering and allow the soil to dry out slightly between waterings.
- Provide adequate light: Pothos plants prefer bright, indirect light but can tolerate low light conditions. If your plant is not getting enough light, move it to a brighter location.
- Fertilize: Pothos plants benefit from regular fertilization. Use a balanced, water-soluble fertilizer and follow the instructions on the label.
- Prune: Yellow leaves can be removed to improve the appearance of the plant. Cut back any yellow leaves to the nearest healthy stem using clean pruning shears.
- Check for pests: Inspect the plant for pests such as mealybugs, spider mites, or scale. If present, treat the plant with insecticidal soap or neem oil according to the product's instructions.
- Repot: If the plant is pot-bound (meaning the roots have filled the container), consider repotting it in a larger container with fresh, well-draining soil.

By following these steps, you may be able to identify and address the underlying cause of the yellowing leaves on your pothos plant.

- **What it establishes:** One mixed sequence can support language, image understanding, and image generation.
- **What remains difficult:** Lossy image tokens, reading text in images, sequential image generation, and detailed visual questions.
- **The next design question:** Can we keep shared context while specializing representations and computation?

Problem Statement: BAGEL

Should understanding and generation share one visual representation?

Motivating example: generate a plausible view of this car's interior

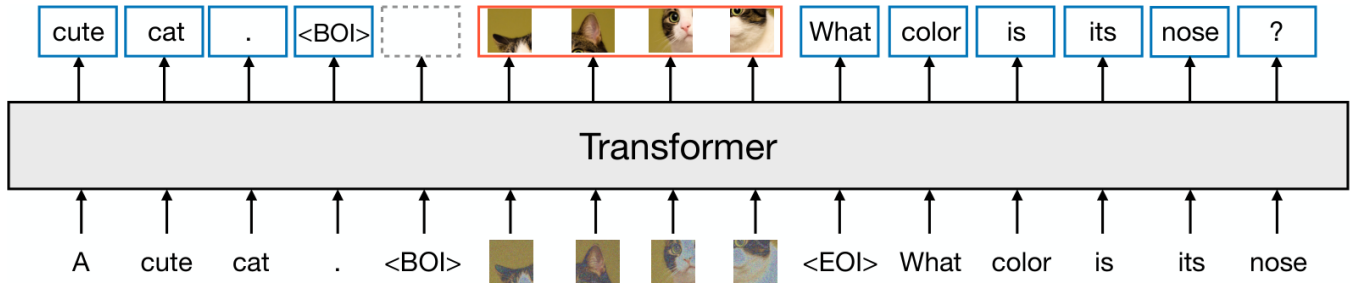
- **A harder request:** Infer a plausible new view, then generate it while preserving relevant scene context.
- **Different visual needs:** Understanding favors semantic features. Synthesis needs detailed appearance information.
- **BAGEL's hypothesis:** Specialize the representations and experts, but retain interaction throughout a shared sequence.

Related Works: BAGEL

Three ways to uniformly combine language modeling and image generation

Family	Representative approach	Notes
Quantized autoregression	Chameleon, Janus	Discrete visual tokenizing; sequential image output (slow)
External generator (e.g., diffusion)	MetaQuery	Strong components; information bottleneck (compress LLM context).
Integrated transformer	Transfusion	Text prediction + continuous image generation (lossless interaction).

Transfusion: an integrated predecessor



Simultaneous use of different objectives.

Georgia Tech

Overview: BAGEL

Autoregressive text + diffusion-based images, using rectified flow

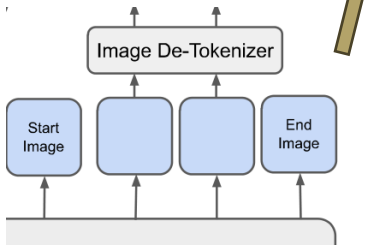
Questions

Make her a Jellycat plush toy.

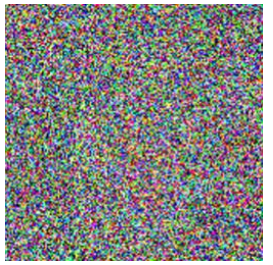
Input

Output

Chameleon



BAGEL



Input

A text prompt, optionally with one or more images

Hybrid generation

Next-token text; iterative image denoising.

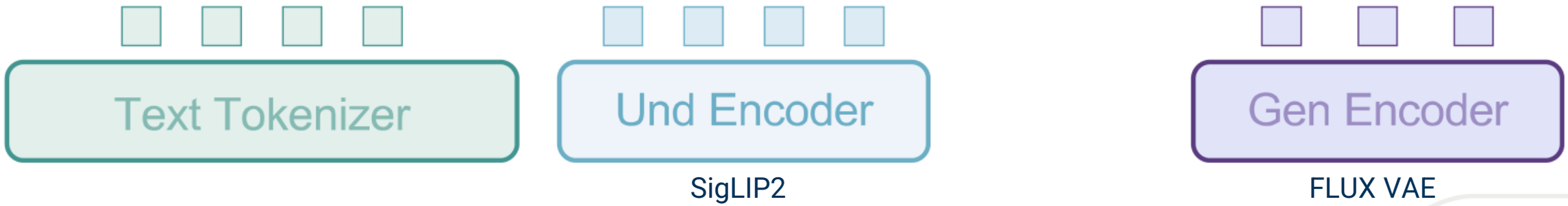
Output

Text answers, new images, or image edits

Approach: BAGEL

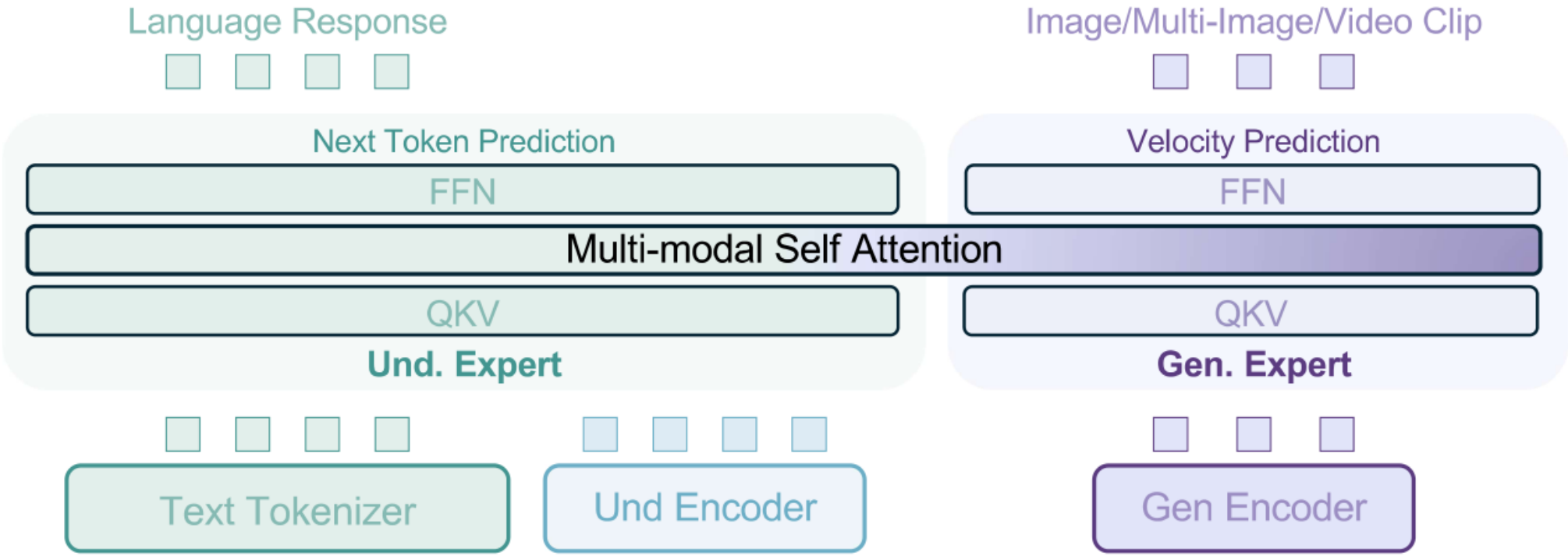
Pretrained Qwen2.5 backbone and two visual representations

- **Language backbone:** Qwen2.5 supplies pretrained language modeling, RMSNorm, SwiGLU, RoPE, and grouped-query attention.
- **Understanding representation:** SigLIP2 features enter through a learned two-layer connector.
- **Generation representation:** A frozen FLUX VAE maps between pixels and continuous image latent tokens. Two types are created per image: clean tokens, and noisy tokens.
- **Why two image encoders:** Semantic understanding and faithful image reconstruction need different information.



Approach: BAGEL

Mixture of transformers with shared attention – 14B total params, 7B active at a time.



Understanding expert

Learns QKV projection weights for text tokens + SigLIP2 features (image understanding tokens)

Generation expert

Learns QKV projection weights for image generation tokens (clean and noisy).

Shared attention at every layer

Concatenate both projections into unified QKV. Execute the attention operation. Split into independent FFN.

Georgia Tech

Approach: BAGEL

Autoregressive text and rectified-flow image generation

Left: straight training paths. Right: learned flow.

- **Text:** Predict the next discrete token with cross-entropy loss.
- **Images:** Learn a velocity field from noisy to clean VAE latent tokens. Minimize velocity MSE.
  - This is the point of the noisy VAE tokens.
- **Joint training:** Loss = 0.25 × text CE + image MSE. The coefficient balances losses, not the fraction of text examples.

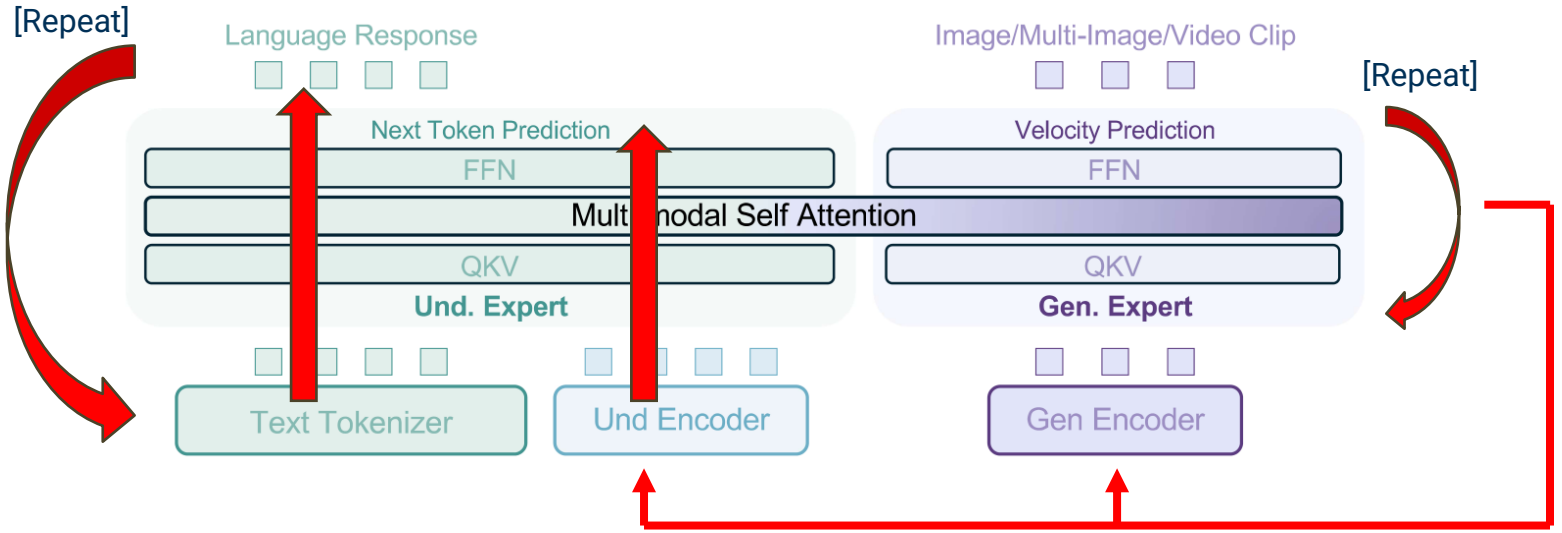
Approach: BAGEL (Inference)

Text Mode

- Autoregressive generation.
  - No noisy VAE tokens.
- Switch mode when an image token is emitted.

Image Generation Mode

1. Turn off understanding expert, but its KV caches are read by Generation expert's attention.
2. Generate Gaussian noise.
3. Repeat:
   - Add timestep embedding
   - Run the whole expert, get velocity prediction, 1 denoise step.
4. FLUX VAE decoding
5. Hand the image to the Understanding expert and add to context as ViT tokens and clean VAE tokens.
6. Switch back to Text Mode.



Mechanisms: Autoregression vs. Flow

Two ways to put an image into a sequence model

Chameleon: discrete next-token codes

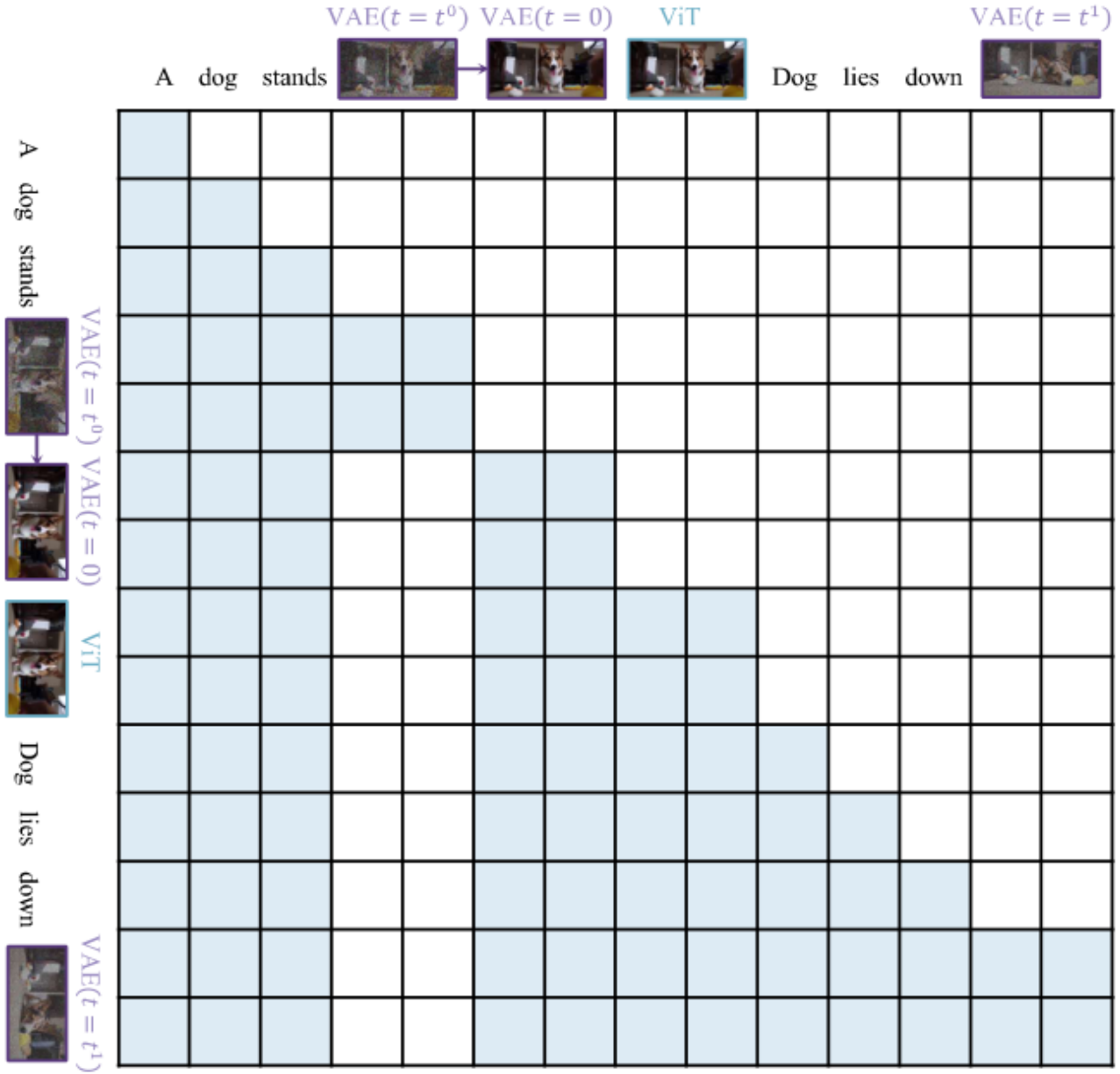
- **Prediction:** one codebook index at a time; 1,024 sequential predictions per image.
- **Training signal:** one cross-entropy objective, the same loss as text.
- **Disadvantage:** quantization limits granularity of understanding/generation, and errors in already-generated patches cannot be fixed.
- **Advantage:** reuse ordinary LLM setup: KV cache, sampling, likelihoods, token-level RL.

BAGEL: rectified flow over latents

- **Prediction:** a velocity for every latent at once, refined over repeated denoising steps.
- **Training signal:** MSE on velocity, weighted against text cross-entropy at 1 : 0.25.
- **Disadvantage:** Manually balancing the losses, extra parameters in memory, and a denoising loop.
- **Advantage:** higher fidelity, bidirectional attention inside an image, classifier-free guidance.

Takeaway: the mechanism controls image quality plus whether the model still has one loss, a likelihood, and an LLM-shaped serving and RL path.

Approach: BAGEL



Causal order across outputs and full attention within images

- **Reading the mask:** Blue cells allow attention. Each row reads the columns it can see.
- Text is causal. Image patches attend bidirectionally.
- **No answer leakage:** A noisy target cannot see its clean copy.
- Noisy tokens cannot be attended to.

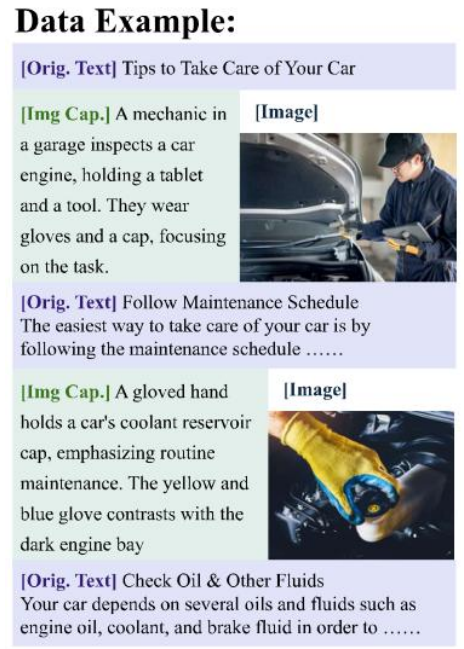
Georgia Tech

Data: BAGEL

How does BAGEL learn from text, image pairs, and interleaved sequences?

Training data pool	Examples	Corpus tokens
Text	400M	0.4T
Paired understanding	500M	0.5T
Paired generation	1,600M	2.6T
Interleaved understanding	100M	0.5T
Video generation sequences	45M	0.7T
Web generation documents	20M	0.4T

Data pools are resampled with changing weights; pretraining and continued training consume about 5.1T tokens.

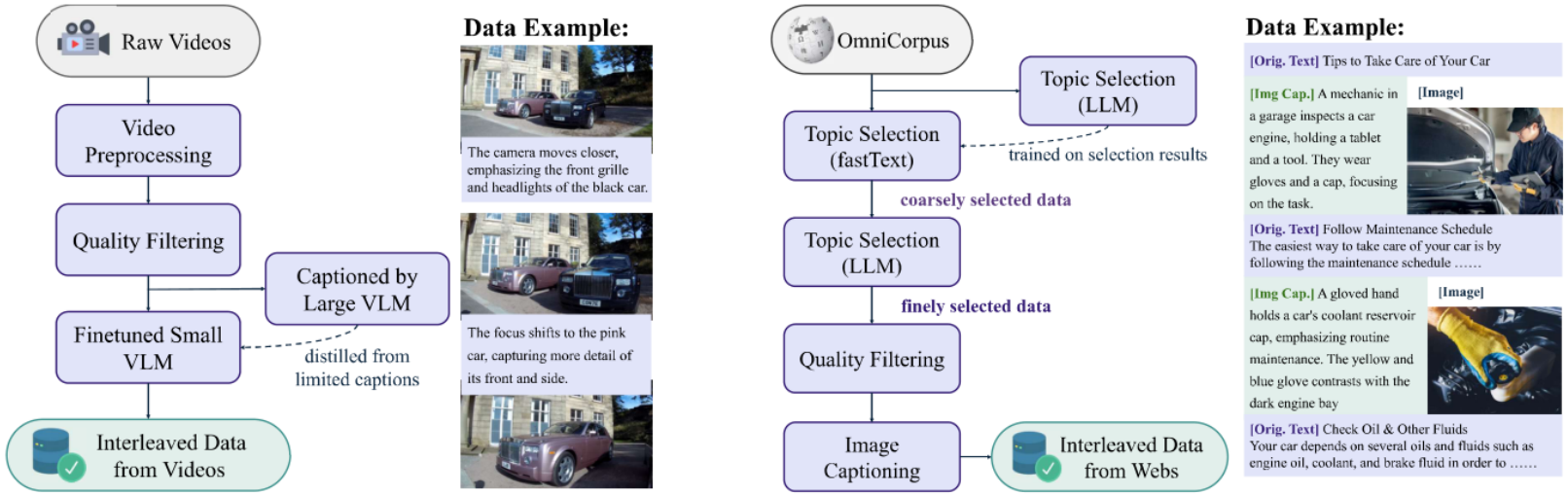


One interleaved web training example

Georgia Tech

Data Construction: BAGEL

Videos and web documents teach relationships across images



(a) Data pipeline for interleaved data from videos. **(b)** Data pipeline for interleaved data from webs.

Video

45M sequences: related frames and descriptions of change

Web

20M documents: captions inserted before images

Reasoning examples

500k examples connect language planning to generation

Training: BAGEL

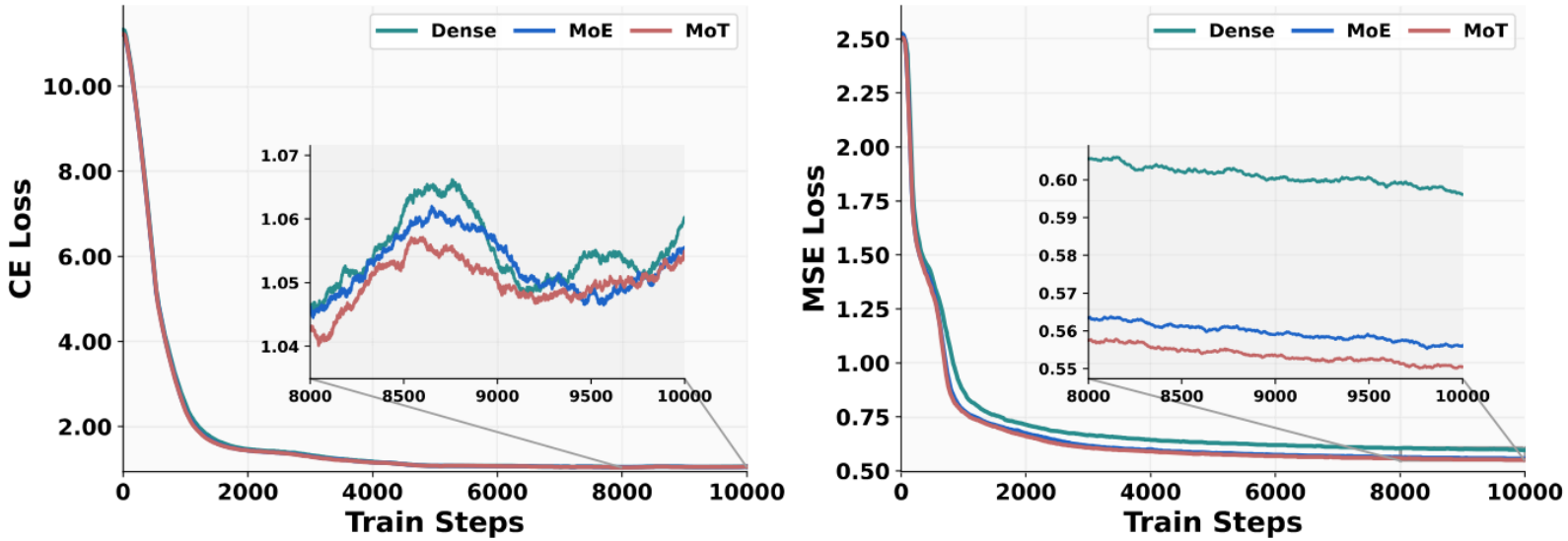
Alignment, pretraining, continued training, and fine-tuning

- **Alignment: 4.9B tokens:** Train the SigLIP2–Qwen connector while both backbones stay frozen.
- **Pretraining: 2.5T tokens:** Train all parameters except the VAE, with 16K context.
- **Continued training: 2.6T tokens:** Increase image resolution, interleaved sampling, and context to 40K.
- **Fine-tuning: 72.7B tokens:** Higher-quality examples. PT / CT / SFT interleaved shares: 25% / 45% / 60%.

Token checkpoints, not the four training stages

Experiments & Results: BAGEL

Controlled architecture ablation



Comparison

- Dense: shared QKV projections and FFN blocks (1 big transformer).
- MoE: shared QKV projection, 2 different FFN blocks.
- MoT: different QKV projections and FFN blocks.

Controlled setup

1.5B backbone. Same data & equal active FLOPs

Result

MoT has the lowest generation loss. Generally strongest text loss. Hyperparameter tuning was needed.

Georgia Tech

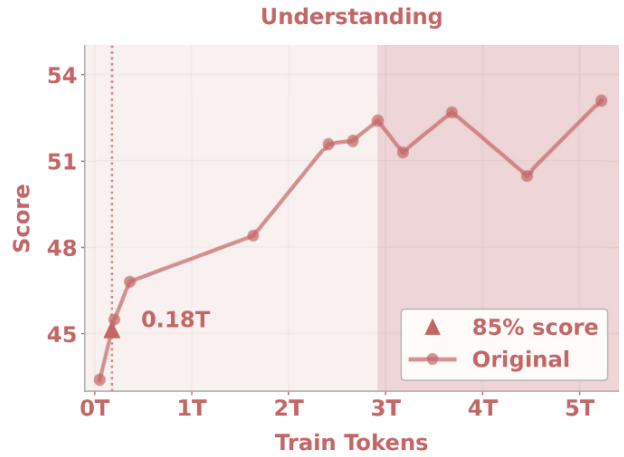
Experiments & Results: BAGEL

Visual understanding benchmarks

Model	MMBench ↑	MMMU ↑	MathVista ↑
Qwen2.5-VL 7B	83.5	58.6	68.2
Janus-Pro 7B	79.2	41.0	—
BAGEL 7B active	85.0	55.3	73.1

MMBench tests general visual questions, MMMU multidisciplinary reasoning, and MathVista visual mathematics. Qwen2.5-VL still leads on MMMU.

Takeaway: learning to generate did not cost understanding. BAGEL beats the unified baseline and leads the specialist on two of three, but it stores 14B parameters to match a 7B model and still trails on MMMU.



Aggregate understanding score across training checkpoints

Georgia Tech

Experiments & Results: BAGEL

Image generation on GenEval

Model / setting	GenEval overall ↑
SD3-medium	0.74
Janus-Pro 7B	0.80
FLUX.1-dev	0.82
BAGEL	0.82
BAGEL + external LLM rewriter	0.88

GenEval tests object-level prompt adherence. Native BAGEL: position 0.64, color attributes 0.63. The 0.88 score adds an external rewriter.

Experiments & Results: BAGEL

Language reasoning before generation and editing

Task / benchmark	BAGEL	+ Self-CoT
Knowledge-based generation: WISE	0.52	0.70
Complex editing: IntelligentBench	44.9	55.3

Without thinking

With thinking

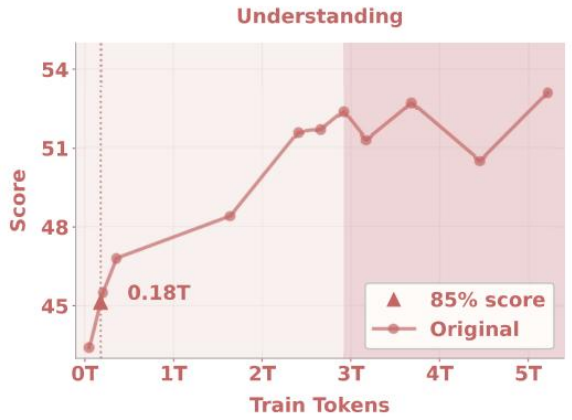
Prompt: “A car made of small cars”

Self-CoT generates a language plan before the image. GEdit English: BAGEL 6.52 vs. Step1X 6.70. IntelligentBench: 350 cases, GPT-4o judge.

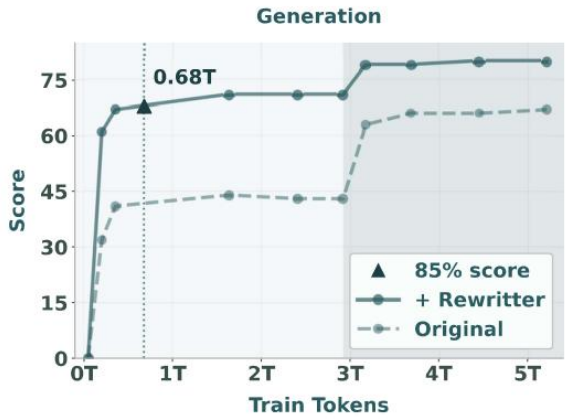
Takeaway: Thinking pays where the gap is knowledge and planning.

Experiments & Results: BAGEL

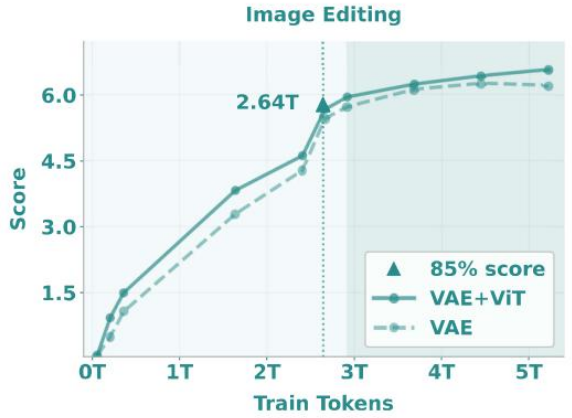
Learning curves across pretraining and continued training



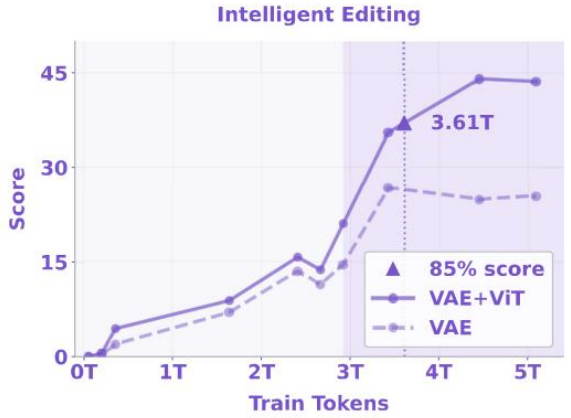
(a) Average score on Image Understanding tasks.



(b) GenEval score on Image Generation task.



(c) GEdit Overall Score on classical Image Editing task.



(d) IntelligentBench Score on Intelligent Editing task

Takeaway: the harder the cross-modal skill, the later it lands.

Tokens to 85% of peak

0.18T understanding
0.68T generation
2.64T editing
3.61T intelligent editing

Semantic context

ViT + VAE helps complex and intelligent editing more than VAE alone.

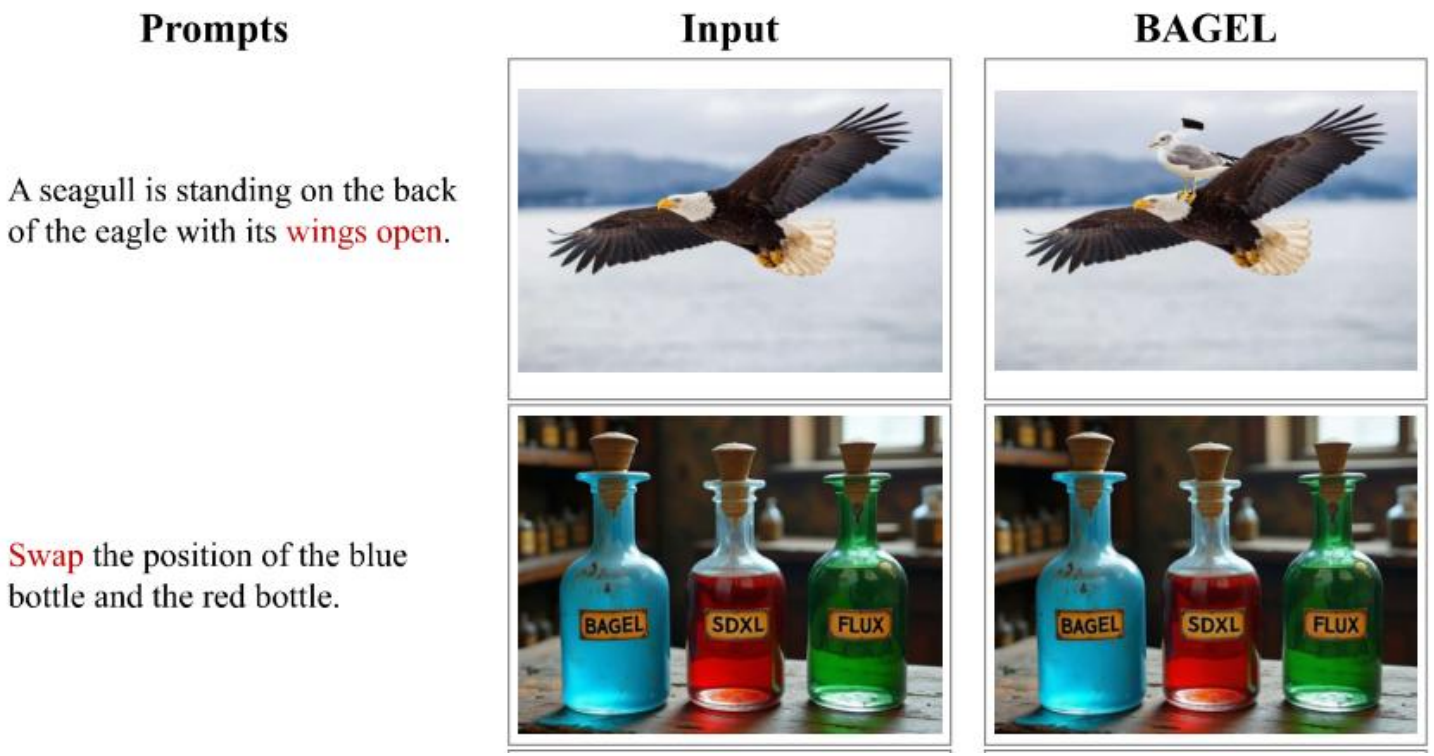
Interpretation

Resolution and the data mixture also change during training.

Georgia Tech

Limitations & Implications: BAGEL

Broader capability does not guarantee precise control

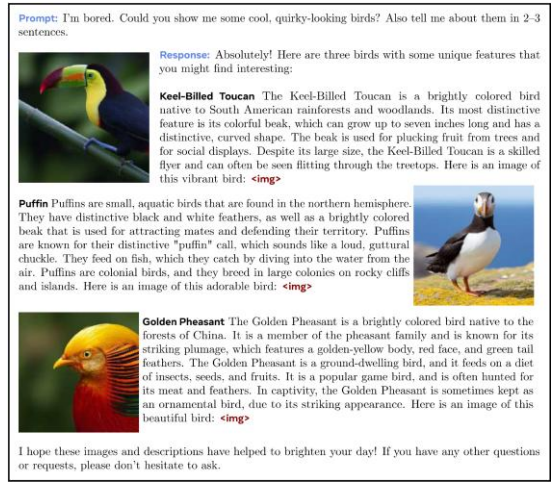


- **A visible failure:** The blue and red bottles should swap positions. BAGEL preserves their order.
- **Other persistent gaps:** Complex text, counterfactual scenes and many interacting objects
- **Practical limits:** Large training budget, model memory, iterative sampling and extra thinking

What Persisted and What Changed?

The shared goal stays; representation and computation diverge

Choice	Chameleon	BAGEL
Image representation	Discrete VQ codes	SigLIP2 + VAE latents
Transformer	Dense shared weights	2 experts + attention
Objective	Next-token loss	Text AR + image flow
Initialization	Train from scratch	Pretrained Qwen2.5
Context	4K tokens	Up to 40K tokens
Shared goal	Interleaved context	Interleaved context



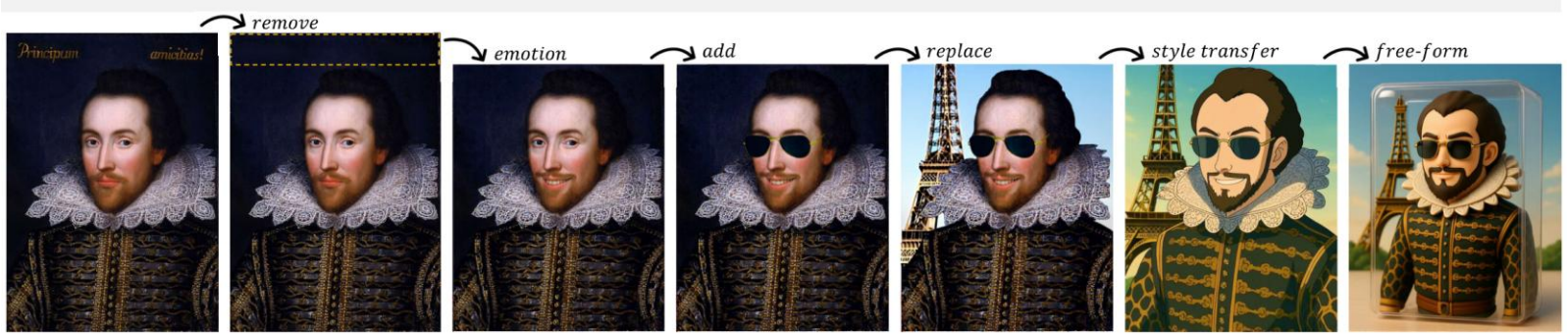
Chameleon: interleaved text and images

BAGEL: contextual image edits

Georgia Tech

Conclusion

What is the ideal unified model?



- **Chameleon:** Native text and image generation through one autoregressive transformer (unified vocabulary).
- **BAGEL:** Shared context with specialized experts, autoregressive text, and rectified-flow images (technically a less unified representation).
- **Evidence:** Broad capabilities coexist; controlled ablations support representation specialization at the tested scale.
- **Remaining limits:** Precise relationships, faithful edits, reliability, and computational cost.