Image Understanding (rec

- ViT
- CLIP
- ImageBind

Vision Transformer (202

- Patchify Image
- Patch Linear Projection
- Image Classification

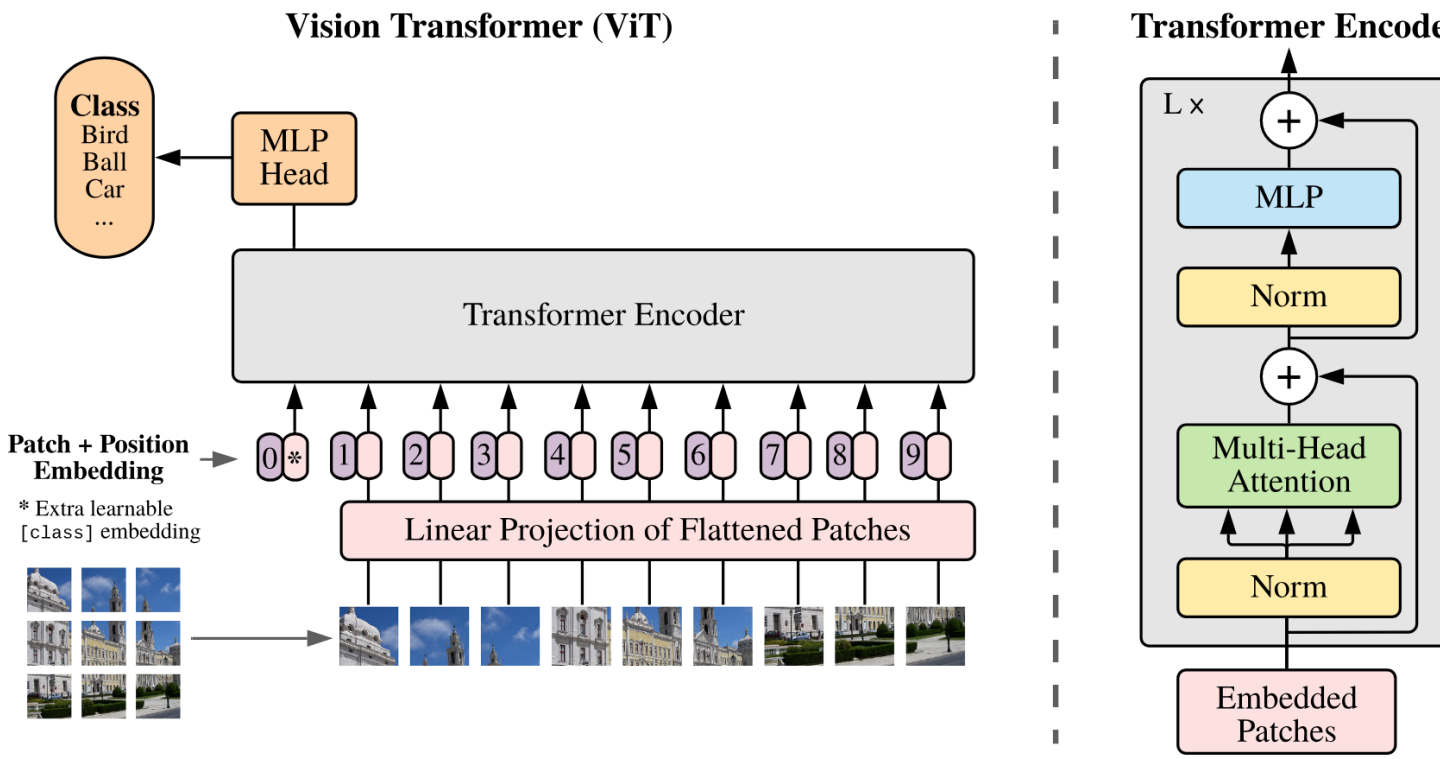


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed eac
add position embeddings, and feed the resulting sequence of vectors to a standard Tr
encoder. In order to perform classification, we use the standard approach of adding an extra
"classification token" to the sequence. The illustration of the Transformer encoder was i
Vaswani et al. (2017).

CLIP (2021)

- ViT Image Encoder
- Text Transformer
- One vector for each image and text
- Text-Image Contrastive Loss

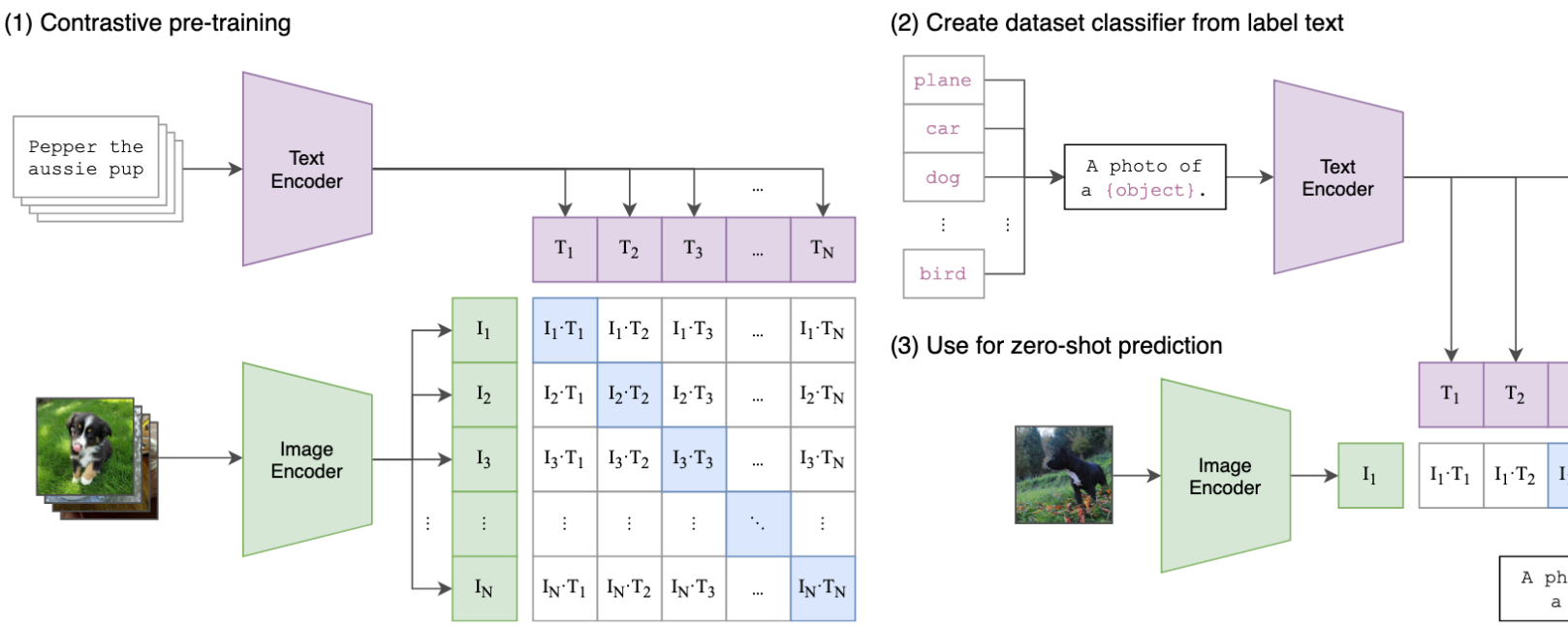


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classi some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descr target dataset's classes.

CLIP (2021)

```
# extract feature representations of each modal
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axi
loss_t = cross_entropy_loss(logits, labels, axi
loss   = (loss_i + loss_t)/2
```

ImageBind (2023)

- Mostly same as CLIP
- More modalities supported

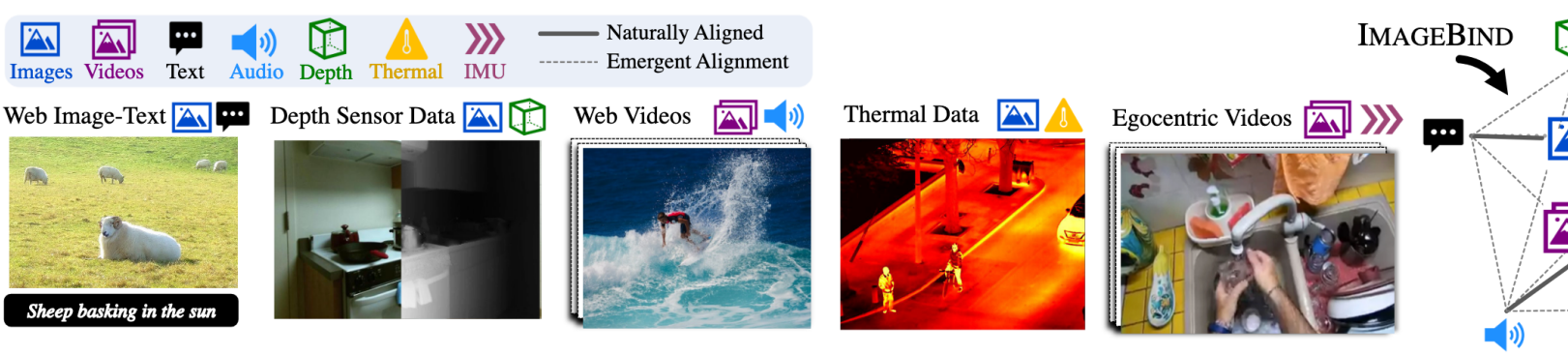


Figure 2. IMAGEBIND overview. Different modalities occur naturally aligned in different data sources, for instance ima video+audio in web data, depth or thermal information with images, IMU data in videos captured with egocentric cameras, BIND links all these modalities in a common embedding space, enabling new emergent alignments and capabilities.

Comparison

Feature	Vision Transformer (ViT)	CLIP	Image
Training Objective	Classification	Image–text contrastive	Multin contra
Text Modality	❌	✅	✅
Image Modality	✅	✅	✅
Audio Modality	❌	❌	✅

Image Understanding (re

- ✅ViT
- ✅CLIP
- ✅ImageBind

Image Understanding

Questions

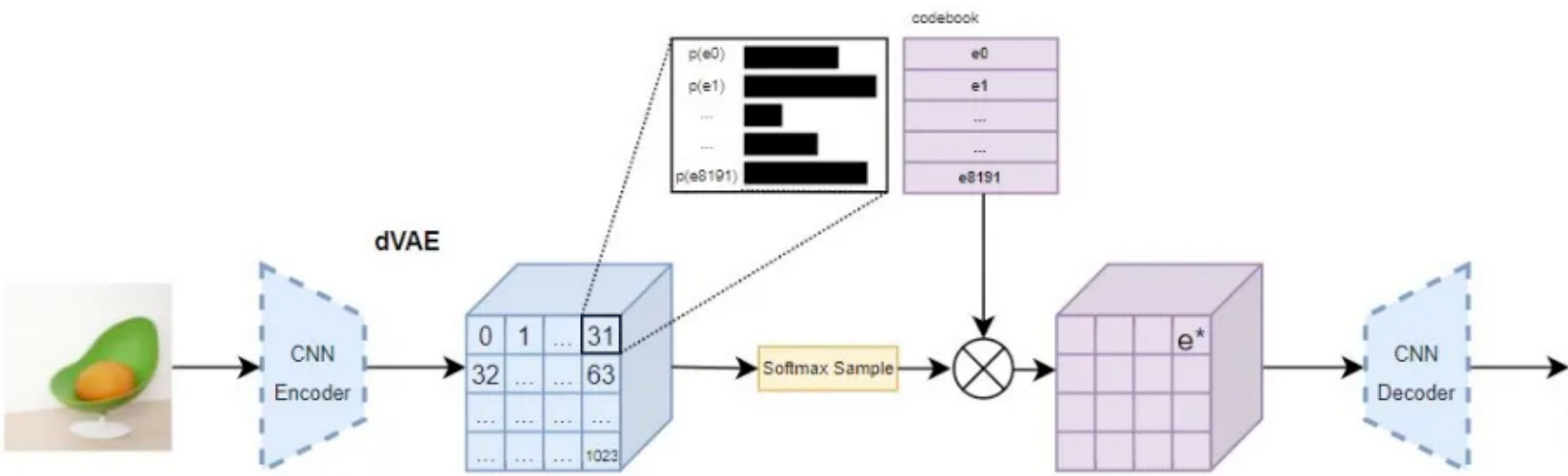
- What advantages does the transformer archi
  bring to image processing compared to CNNs
- How does CLIP's ability to learn from natural
  language supervision make it more flexible th
  traditional image classification models ❓
- What is shared embedding space for differen
  modalities ❓

Image Generation (not LL based)

- Dalle
- Dalle2
- VAR

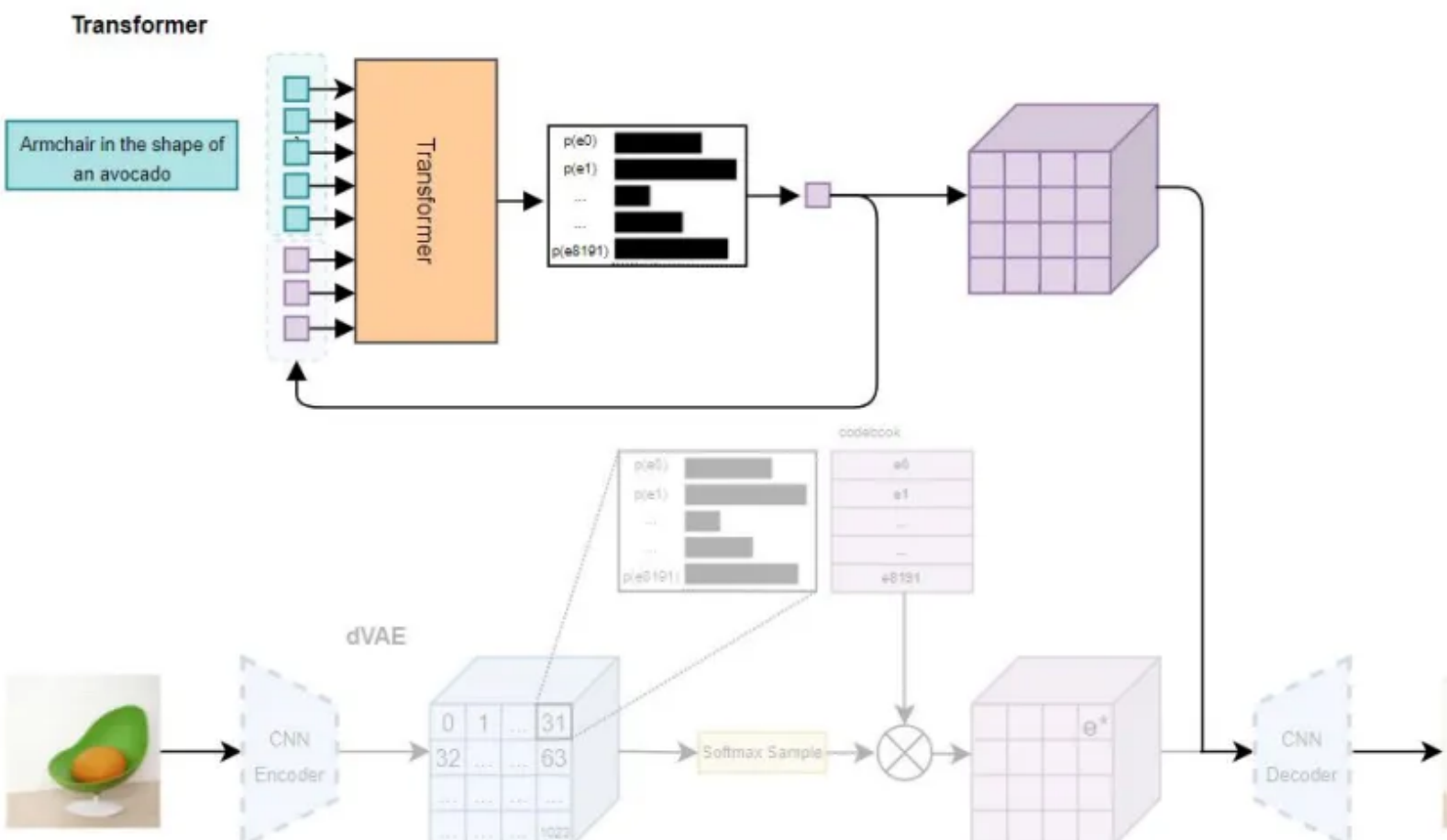
Dalle (2021)

- Compress Image to Discrete Latent Space
- Transformer Causal Modeling Of Text and Im



Dalle (2021)

- Compress Image to Discrete Latent Space
- Transformer Causal Modeling Of Text and Im



Dalle2 (2022)

- Pretrained CLIP (Joint Embedding Space)
- Prior Model for Text to Image embedding Tra
- Diffusion Decoder

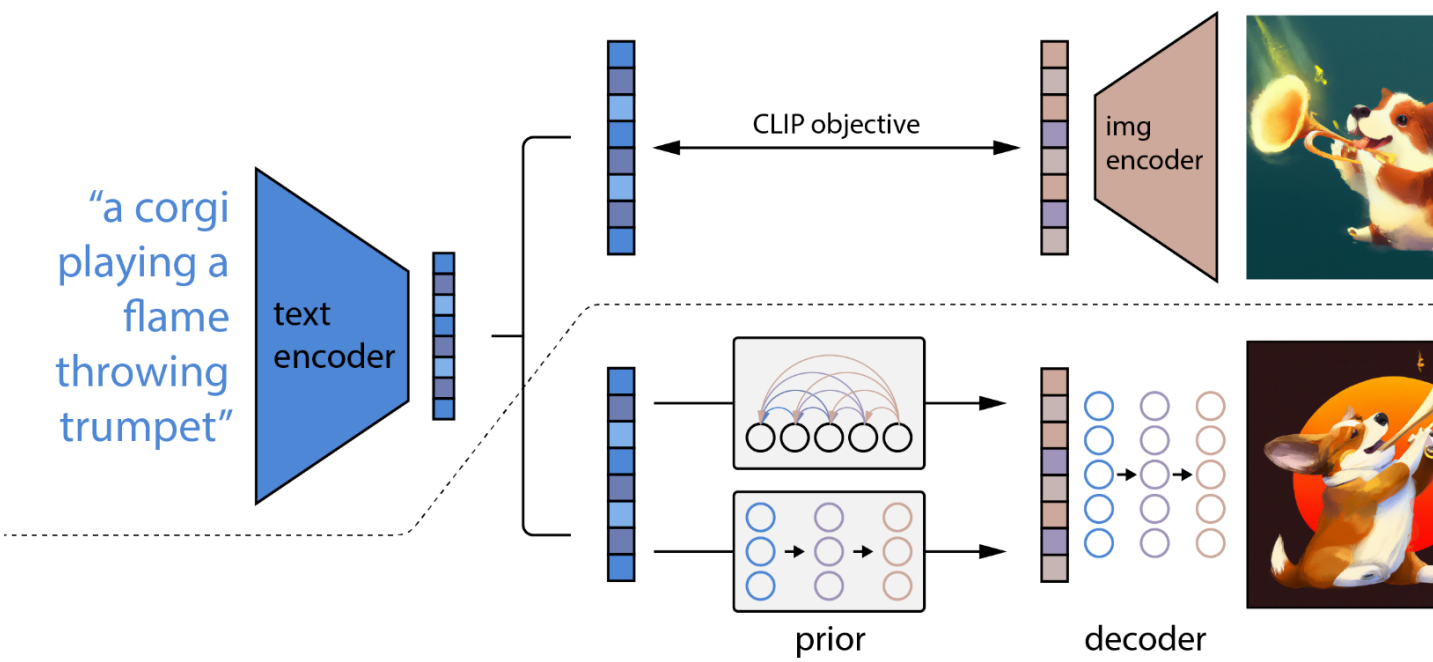


Figure 2: A high-level overview of unCLIP. Above the dotted line, we depict the CLIP traini
through which we learn a joint representation space for text and images. Below the dotted line, w
text-to-image generation process: a CLIP text embedding is first fed to an autoregressive or dif
to produce an image embedding, and then this embedding is used to condition a diffusion dec
produces a final image. Note that the CLIP model is frozen during training of the prior and deco

VAR (2024)

- Visual Autoregressive Modeling via Next-Sca Prediction
- VQVAE for Image Compression

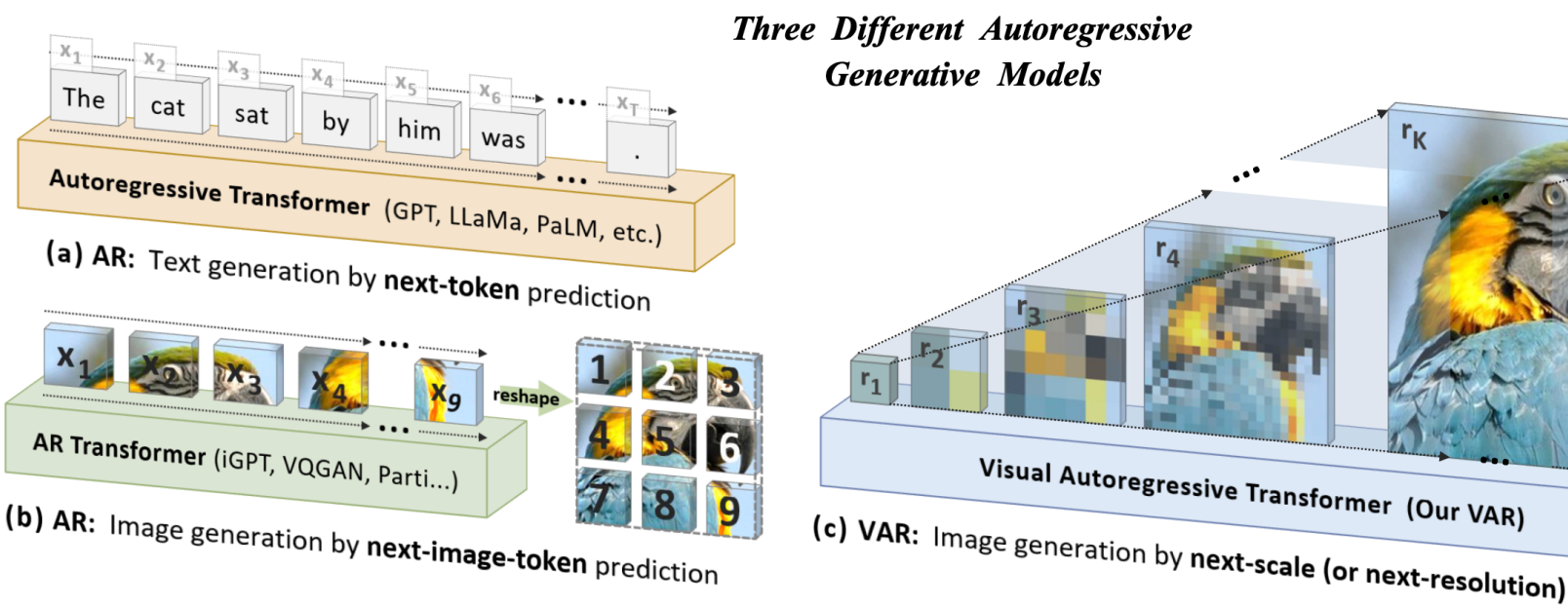


Figure 2: **Standard autoregressive modeling (AR) *vs.* our proposed visual autoregressive model** (a) AR applied to language: sequential text token generation from left to right, word by word; (b) A to images: sequential visual token generation in a raster-scan order, from left to right, top to botton for images: multi-scale token maps are autoregressively generated from coarse to fine scales (lowe resolutions), with parallel token generation within each scale. VAR requires a multi-scale VQVAE t

Image Generation

- ✅Dalle - Autoregressive image generation
- ✅Dalle2 - Diffusion image generation
- ✅VAR - Visual Autoregressive Modeling via Scale Prediction

Questions

- Dalle{1,2} differences ❓
- How to teach LLMs to generate images ❓

LLM-based Approaches

Understanding

- Early Fusion - Inputs Concatenation
- Deep Fusion - Cross Attention

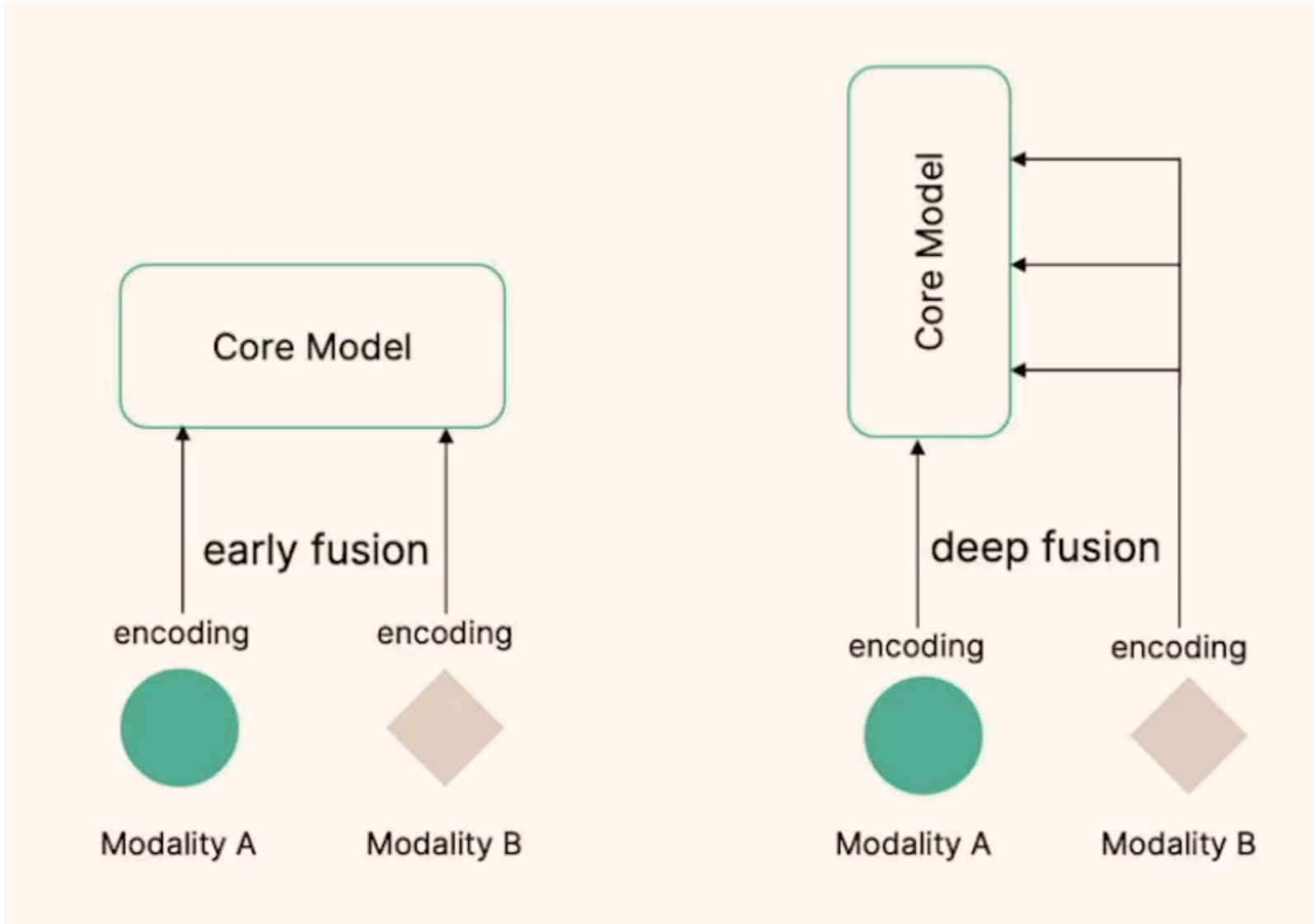


Image Credit

Модели:

- Flamingo
- Formage
- Encodec (2022)
- Llava

Flamingo (2022)

- CLIP Image Encoder
- ❗ Interleaved Data
- ❗ Deep Fusion

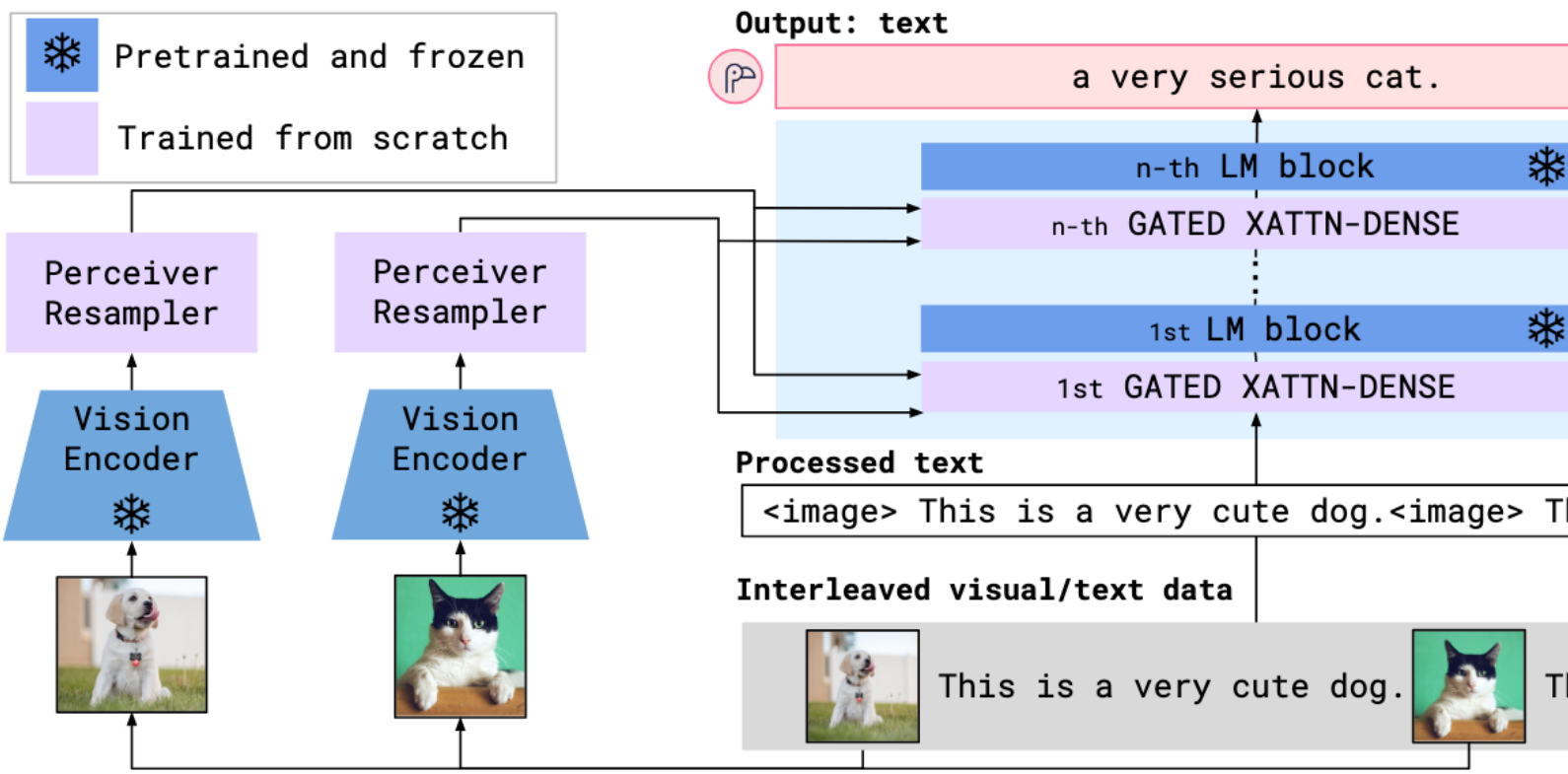
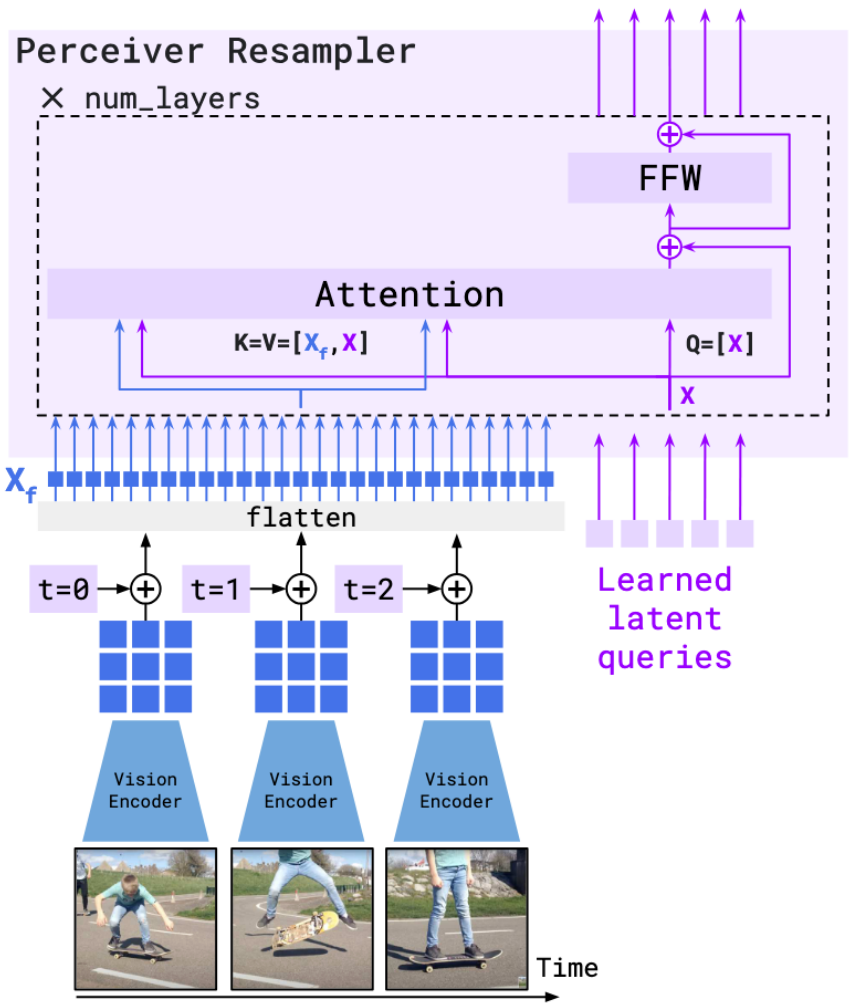


Figure 3: **Flamingo architecture overview.** Flamingo is a family of visual language model that take as input visual data interleaved with text and produce free-form text as output.

Flamingo (2022)

❗ Perceiver Resampler

- Fixed image embeddings size
- Learned Query Vectors




```python
def perceiver_resampler(
    x_f,  # The [T, S, d] visual features (T=time, S=sp
    time_embeddings,  # The [T, 1, d] time pos embeddin
    x,  # R learned latents of shape [R, d]
    num_layers,  # Number of layers
):
  """The Perceiver Resampler model."""

  # Add the time position embeddings and flatten.
  x_f = x_f + time_embeddings
  x_f = flatten(x_f)  # [T, S, d] -> [T * S, d]
  # Apply the Perceiver Resampler layers.
  for i in range(num_layers):
    # Attention.
    x = x + attention_i(q=x, kv=concat([x_f, x]))
    # Feed forward.
    x = x + ffw_i(x)
  return x
```

Figure 5: **The Perceiver Resampler** module maps a *variable* size grid of spatio-tempo features output by the Vision Encoder to a *fixed* number of output tokens (five in the figu pendently from the input image resolution or the number of input video frames. This tra has a set of learned latent vectors as queries, and the keys and values are a concatenati spatio-temporal visual features with the learned latent vectors.

Fromage (2023)

- CLIP Image Encoder
- Opt LLM
- Early Fusion
- ❗Image Retrieval

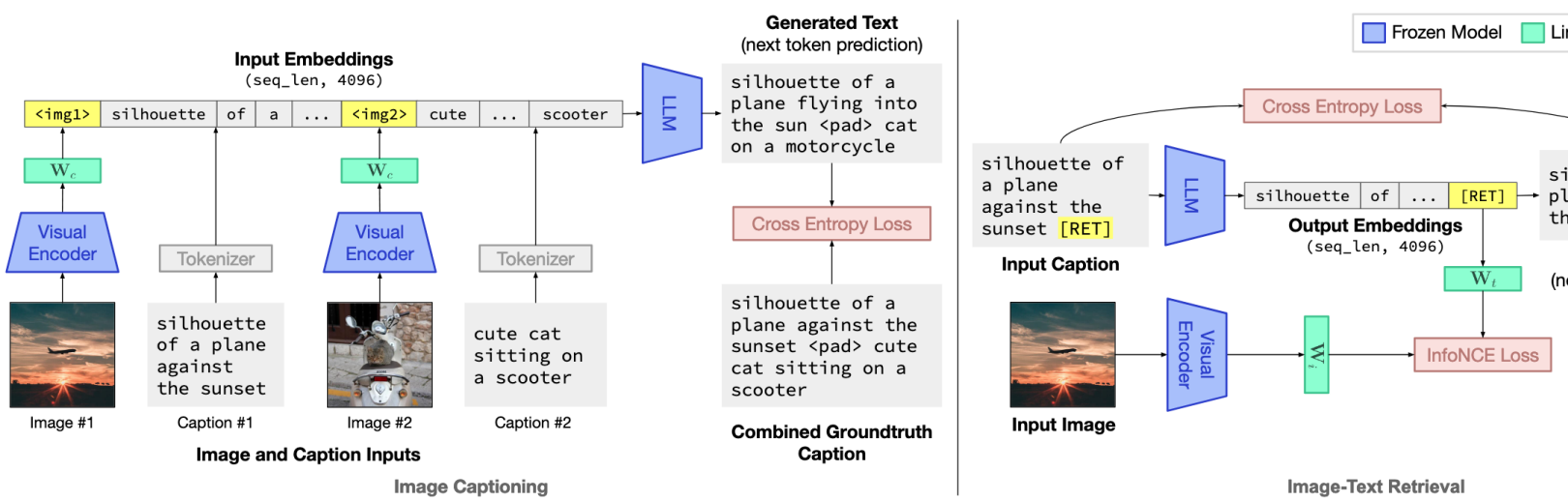


Figure 2. Overview of the FROMAGe architecture. FROMAGe is a model trained on image-text pairs for image captioning a retrieval. It is capable of processing arbitrarily interleaved image and text inputs, and producing interleaved images and te

BLIP2 (2023)

- CLIP Image Encoder
- Opt LLM / Flan T5
- Early Fusion
- ❗QFormer for Image Embedding
- Freezed LLM

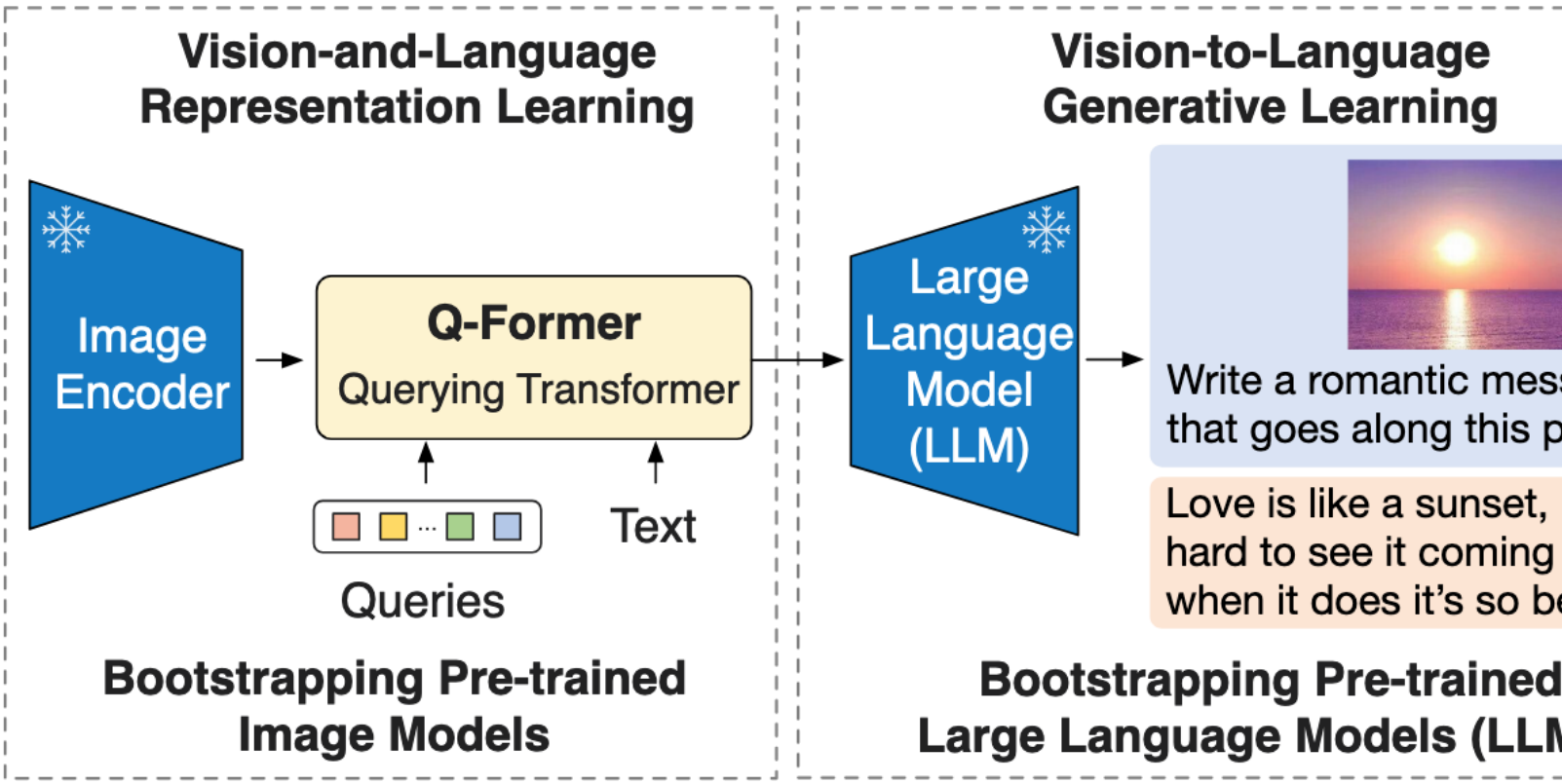


Figure 1. Overview of BLIP-2's framework. We pre-tr lightweight Querying Transformer following a two-stage egy to bridge the modality gap. The first stage bootstraps v language representation learning from a frozen image encode second stage bootstraps vision-to-language generative lea from a frozen LLM, which enables zero-shot instructed ima text generation (see Figure 4 for more examples).

Llava (2023)

- CLIP Image Encoder
- Vicuna-7B LLM
- 2-stage training:
  - Feature Alignment
  - End-to-End SFT

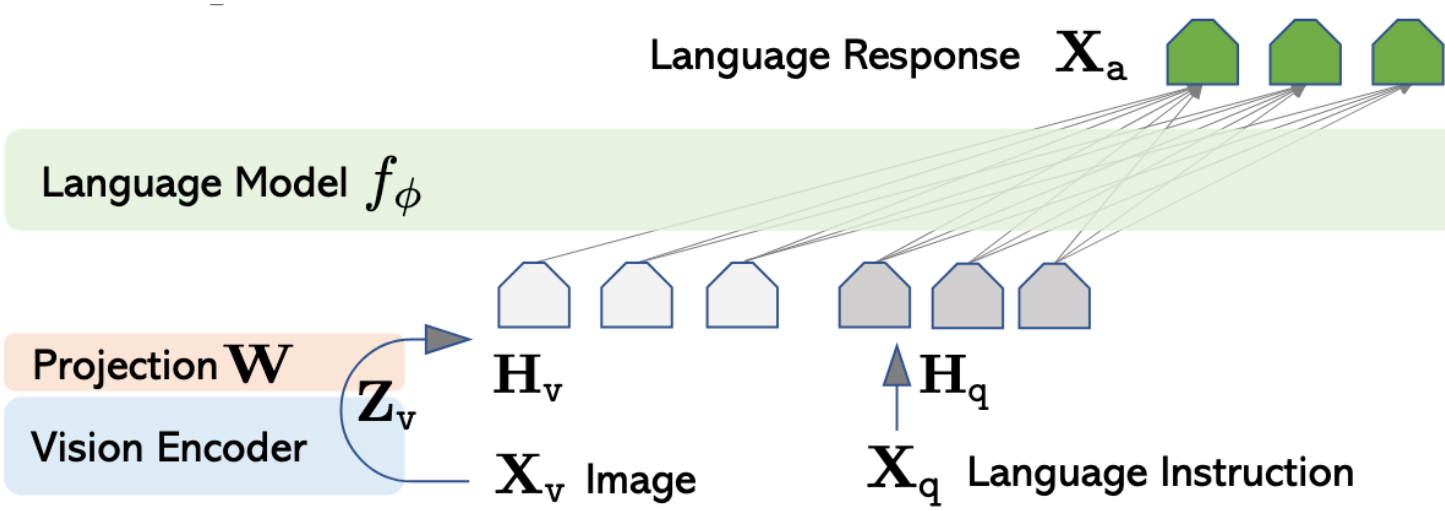


Figure 1: LLaVA network architecture.

LLM Image Understandin

Comparison

Feature / Model	Flamingo	Fromage	BLIP2	LL
Fusion Type	Deep	Early	Early	Ea
Interleaved Data	✅	❌	❌	❌
Image Retrieval	❌	✅	✅	❌
Instruction Tuning	❌	❌	✅	✅

LLM Image Understandin

Questions

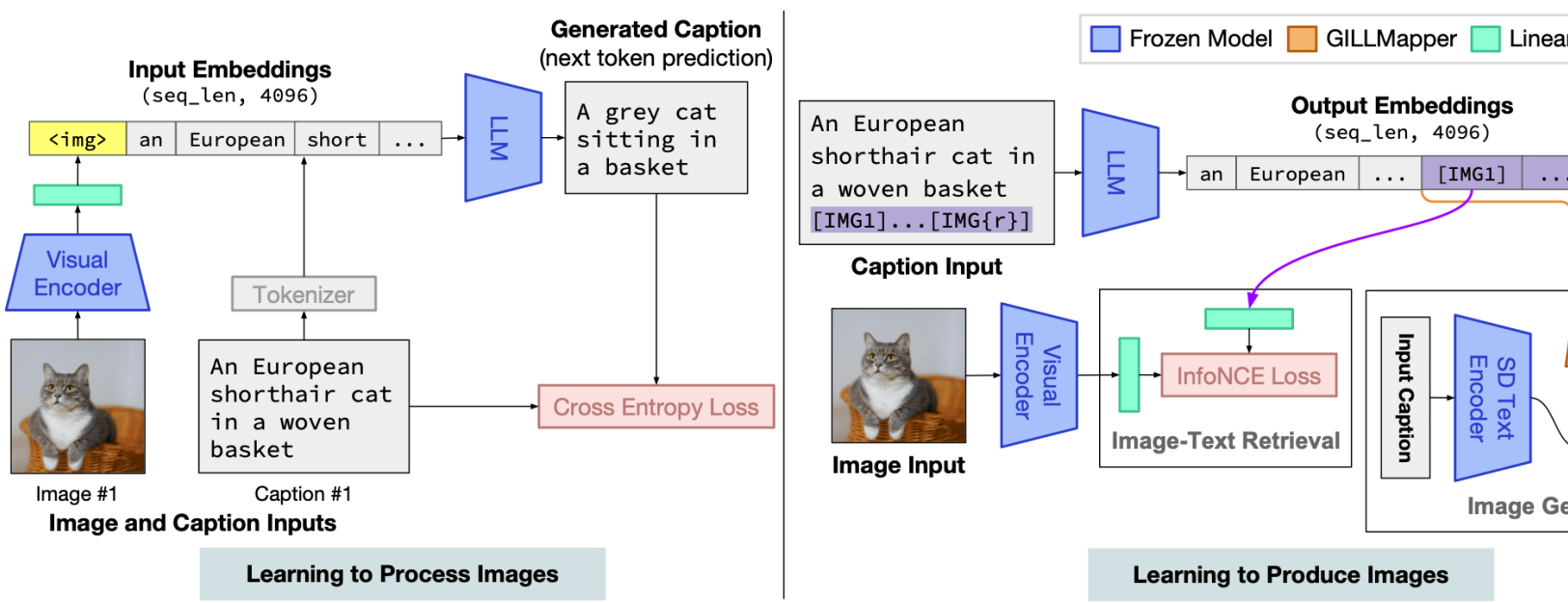
- Why **CLIP** ViT is utilized in most MLLM mode
- DeepFusion and Encoder-Decoder Transform
  Difference ❓
- Flamingo Perceiver Resampler vs Blip2 QFor

LLM Image Generation / Editing

- GILL
- Mgie
- Bagel

GILL (2023)

- Image Understanding
- Image Retrieval and Generation
- LLM learns to generate image embeddings fo generation model
- 8 visual tokens



GILL (2023)

- How GillMapper looks like ❓

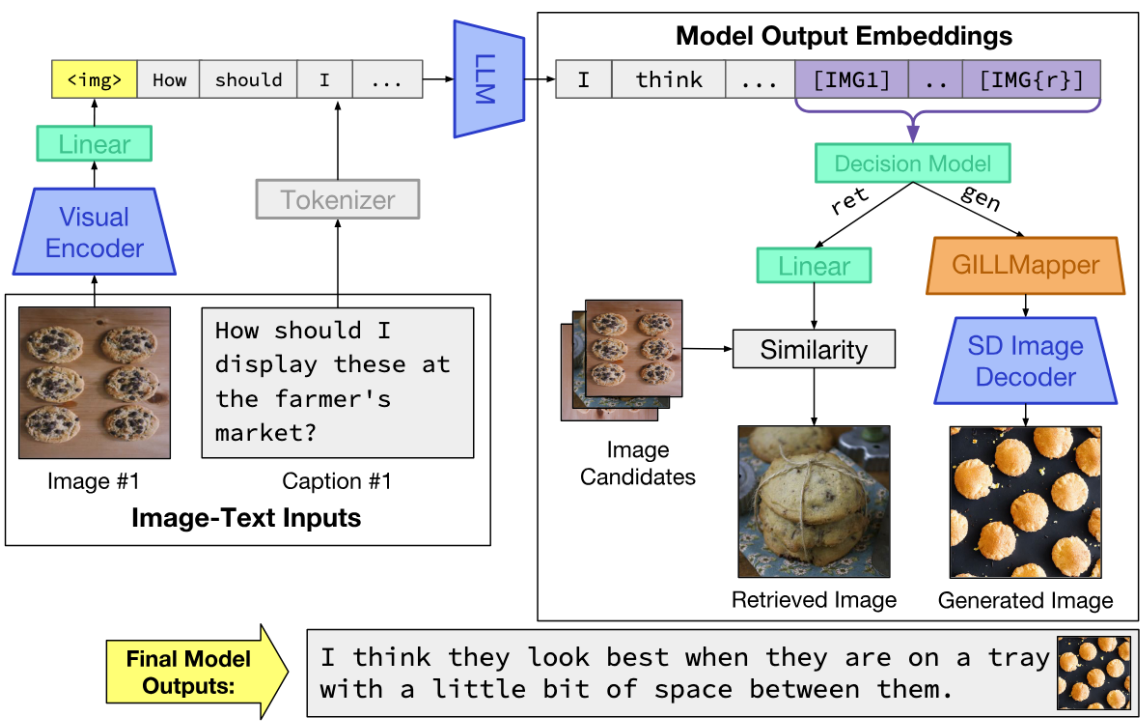


Figure 3: Inference time procedure for GILL. The model takes in image and text inputs, and produces text interleaved with image embeddings. After deciding whether to retrieve or generate for a particular set of tokens, it returns the appropriate image outputs.

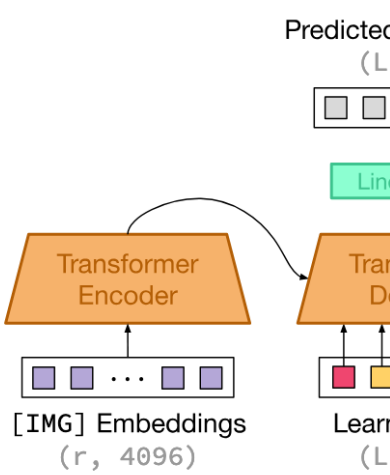
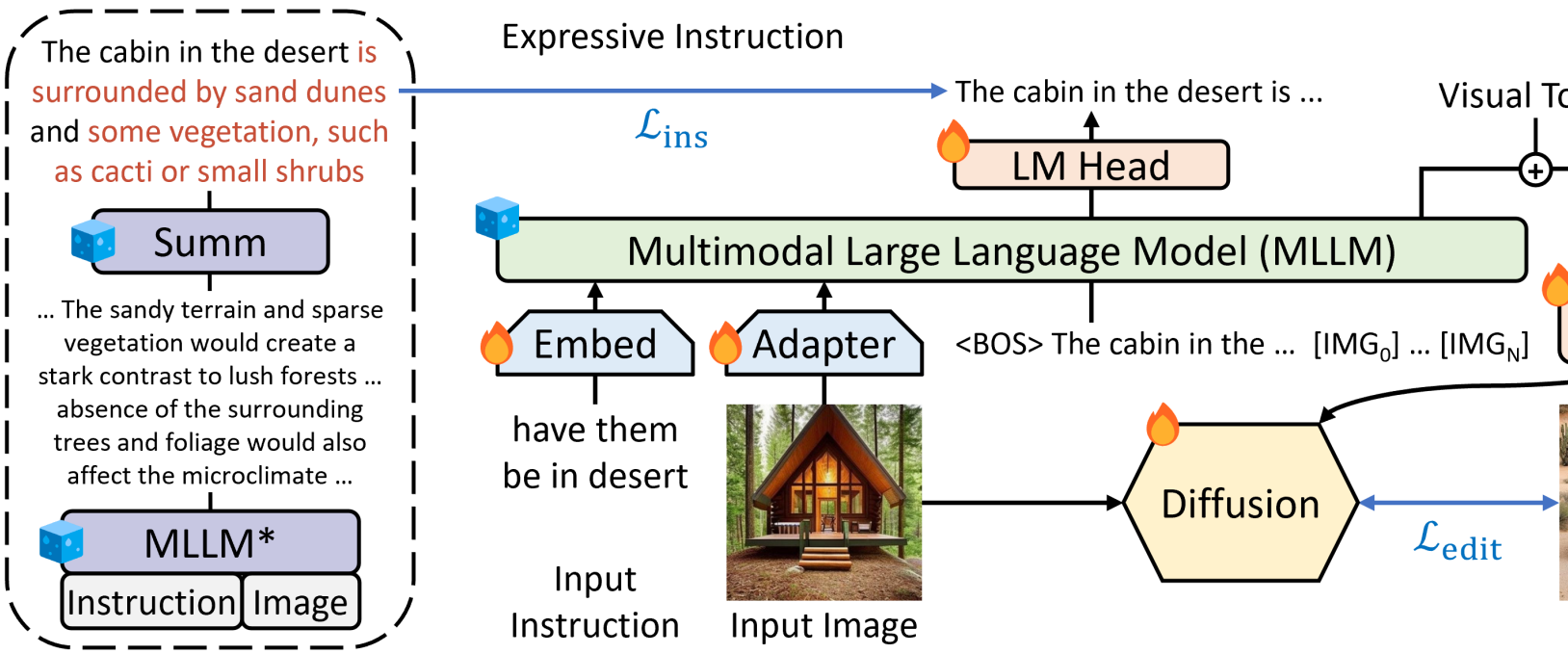


Figure 4: GILLMapp
architecture. It is co
on the hidden [IMG]
tations and a sequenc
query embedding vec

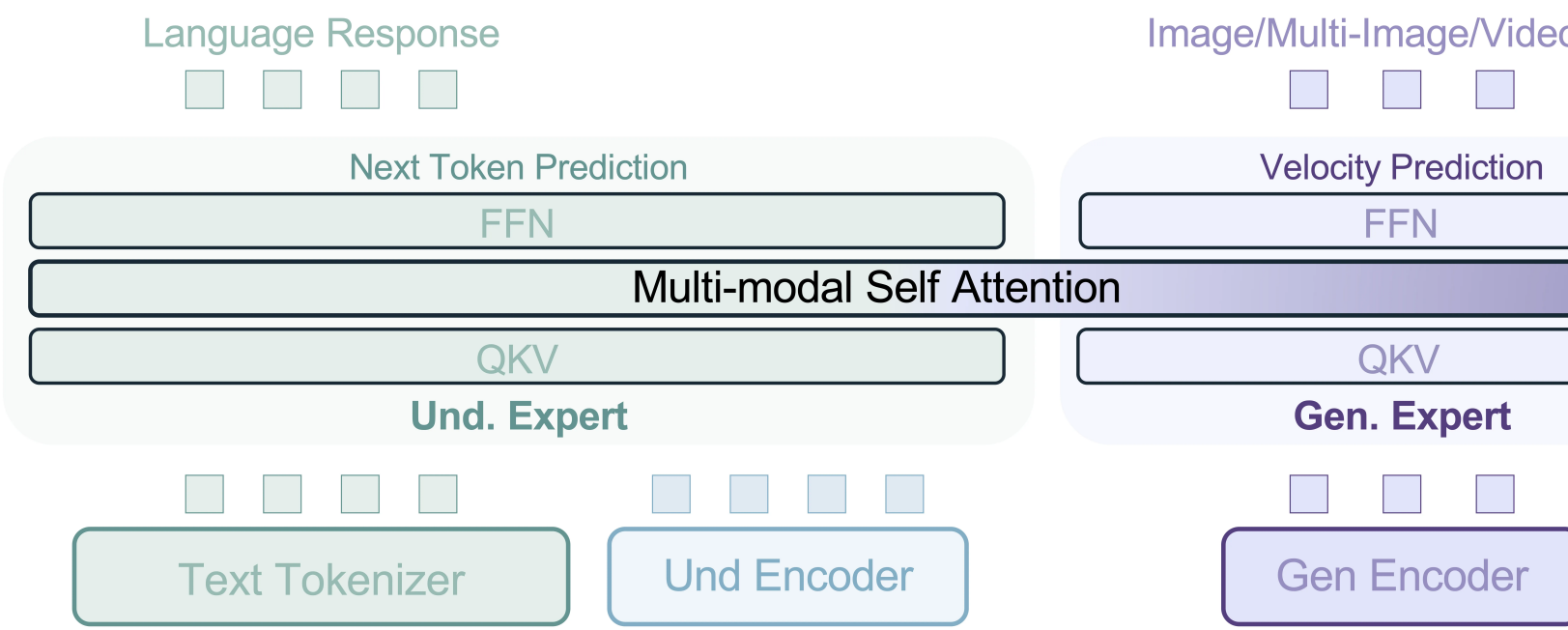
Mgie (2024) Apple

- ❗ Task: Instruction-based editing
- Expressive Instructions
- Mostly inspired by GILL



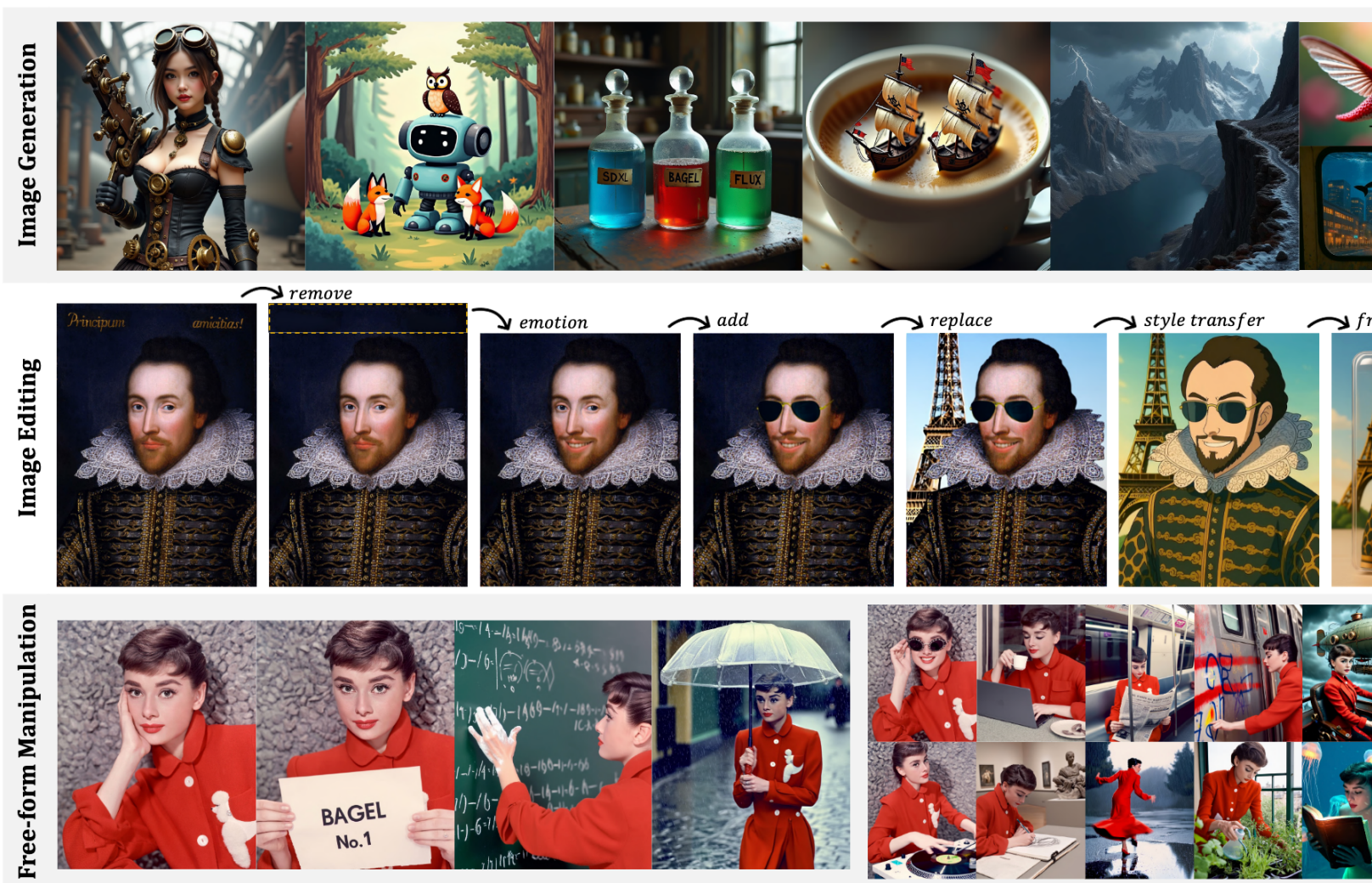
Bagel (2025) ByteDance

- ❗ Tasks: Understanding, Editing, Style Transf
- Thinking before Generation
- Data
- Qwen2.5 LLM
- SigLIP2 Image Encoder
- FLUX VAE


Language Response
Image/Multi-Image/Video
Next Token Prediction
Velocity Prediction
FFN
FFN
Multi-modal Self Attention
QKV
QKV
Und. Expert
Gen. Expert
Text Tokenizer
Und Encoder
Gen Encoder

Bagel (2025) ByteDance

Samples



LLM Image Generation / Editing

- ✅GILL
- ✅Mgie
- ✅Bagel

LLM Image Generation / Editing

Metrics

- Side-by-side User Study

Materials

- Елизавета Гончарова | Мультимодальные подходы и LLM
- Максим Куркин | MLLMs
- Обзор на русском от Максима мультимод подходов
- Molmo - VLLM от allenai с открытым кодом тренировки
- InternVL-2.5 - SOTA VLLM Image Encoder
- Malvina - как решают задачу pixel-perfect редактирования изображений в команде Gigachat

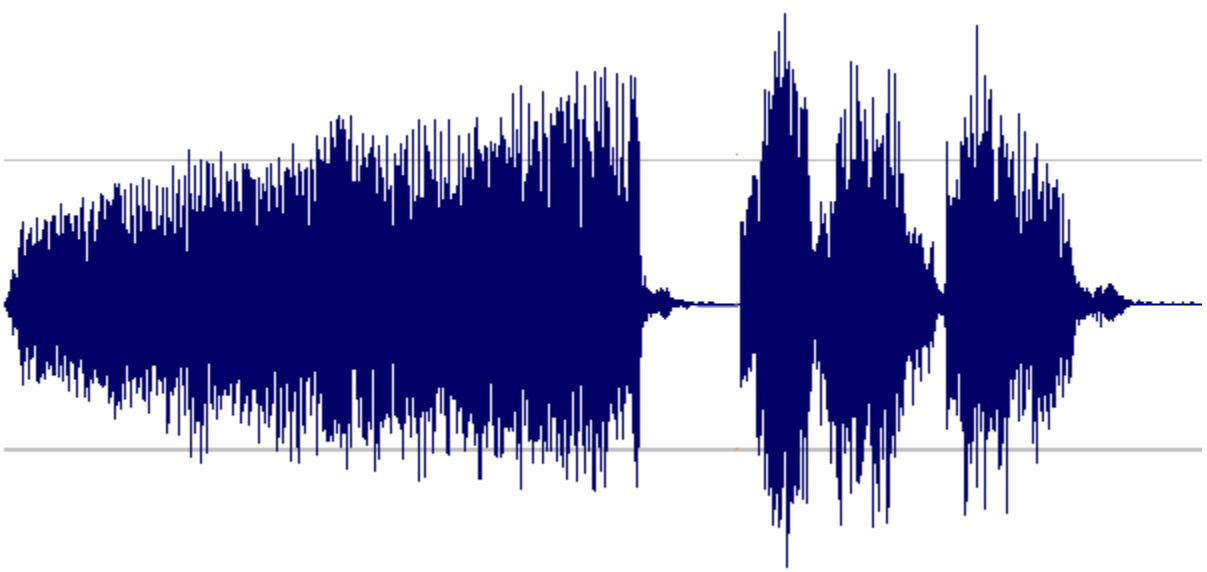
Audio Modality

Audio Representations

- Waveform
- Mel-Spectrogram

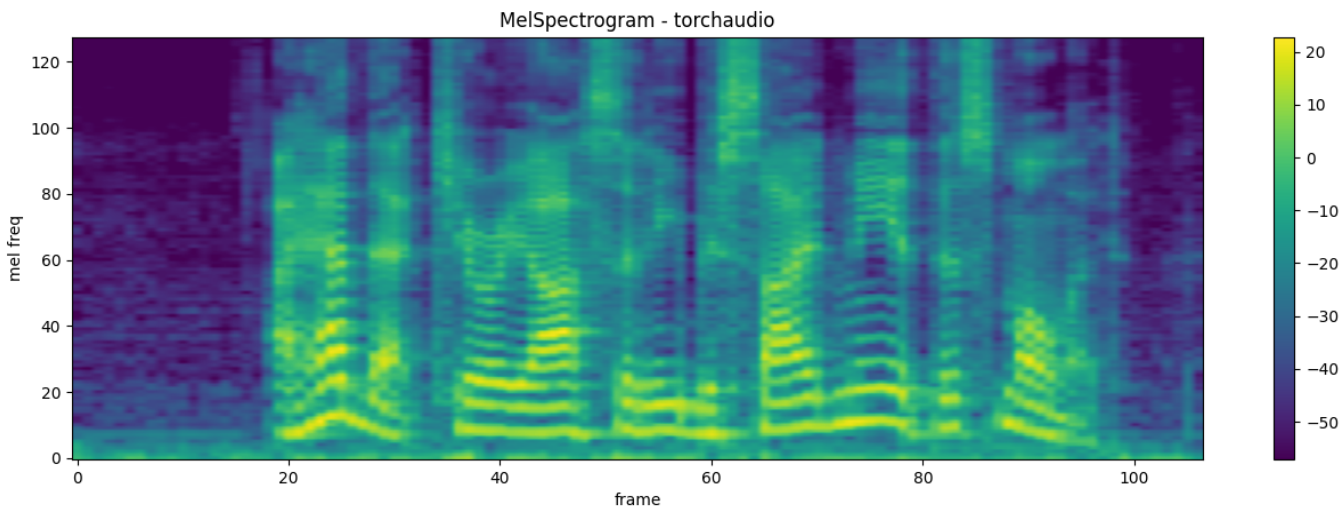
Waveform

- Raw audio samples
- Time series
- Sample rate: 16kHz-48kHz



Mel-Spectrogram

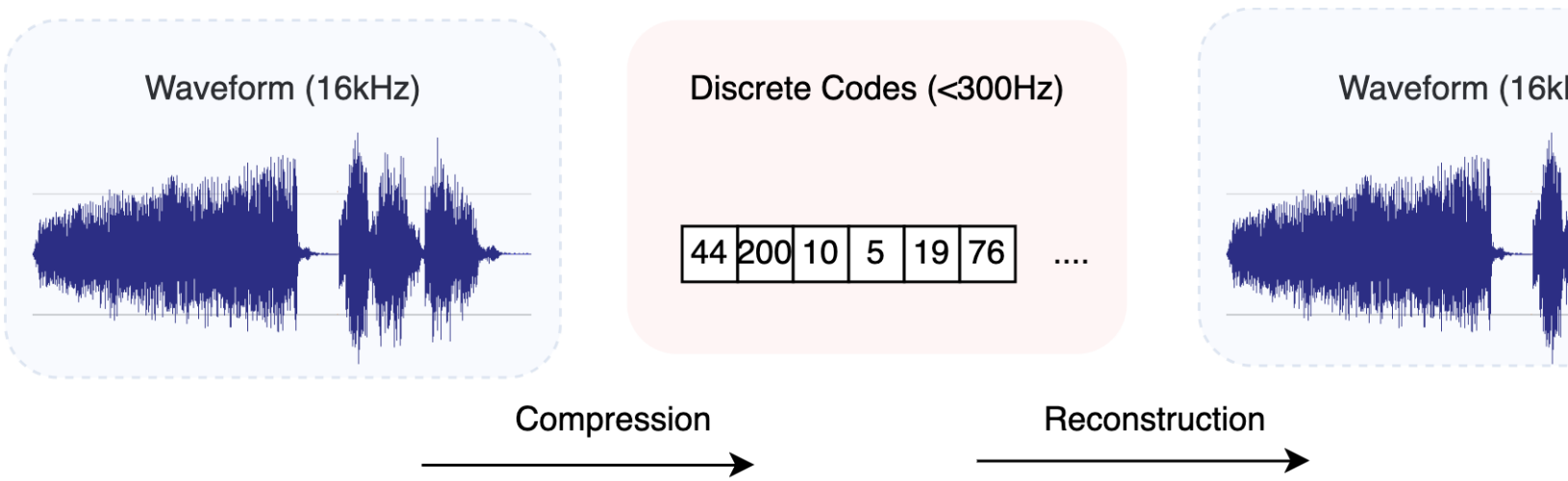
- Semantically compressed (but approximately size in bytes as waveform)
- Windowed Fourier Transform
- Hop size = 10 ms (Time frames)
- Mel bins = 40 to 128 (Frequency bins)



Audio Codecs

🎯Goal:

Discrete Audio Representation for Causal Mod
(and/or audio compression)



Audio Codecs

🤖**Models:**

- Encodec (2022)
- Mimi (2024)

Encodec (2022)

Architecture

- Varying Bitrates (Residual Vector Quantizati

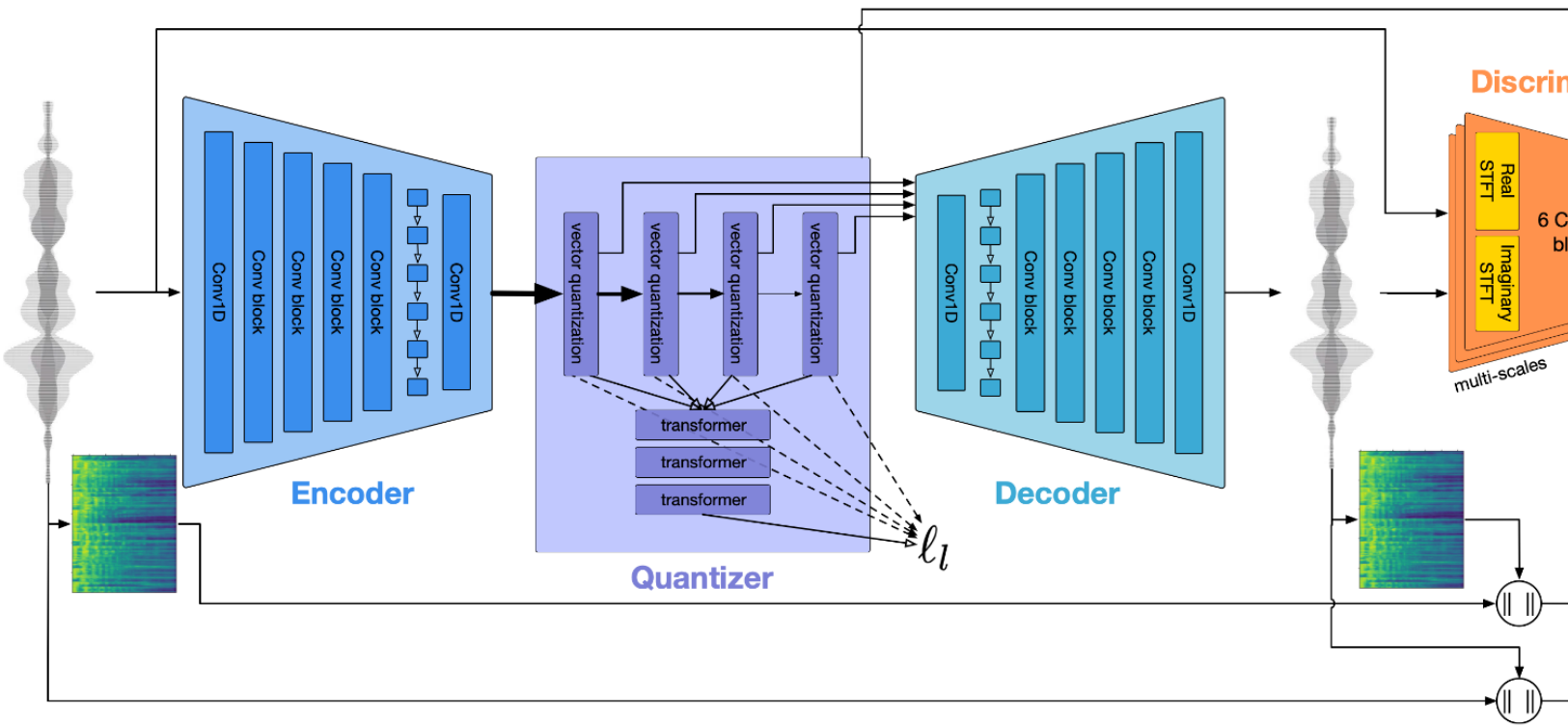


Figure 1: EnCodec : an encoder decoder codec architecture which is trained with reconstructi
ℓ_t) as well as adversarial losses (ℓ_g for the generator and ℓ_d for the discriminator). The resic
quantization commitment loss (ℓ_w) applies only to the encoder. Optionally, we train a small T
language model for entropy coding over the quantized units with ℓ_l, which reduces bandwidth eve

Encodec (2022)

Loss

Reconstruction loss

Reconstruction Loss. The reconstruction loss term is comprised of a time and a frequency do
term. We minimize the L1 distance between the target and compressed audio over the time do
$\ell_t(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_1$. For the frequency domain, we use a linear combination between the L1 and
over the mel-spectrogram using several time scales (Yamamoto et al., 2020b; Gritsenko et al., 2020).

$$\ell_f(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{|\alpha| \cdot |s|} \sum_{\alpha_i \in \alpha} \sum_{i \in e} \|\mathcal{S}_i(\boldsymbol{x}) - \mathcal{S}_i(\hat{\boldsymbol{x}})\|_1 + \alpha_i \|\mathcal{S}_i(\boldsymbol{x}) - \mathcal{S}_i(\hat{\boldsymbol{x}})\|_2,$$

Encodec (2022)

Loss

Discriminative Loss

The adversarial loss for the generator is constructed as follows, $\ell_g(\hat{\boldsymbol{x}}) = \frac{1}{K}\sum_k \max(0, 1 - D_k(\hat{\boldsymbol{x}}))$, v
the number of discriminators. Similarly to previous work on neural vocoders (Kumar et al., 2019; K
2020; You et al., 2021), we additionally include a relative feature matching loss for the generator.

$$\ell_{feat}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{KL}\sum_{k=1}^{K}\sum_{l=1}^{L}\frac{\|D_k^l(\boldsymbol{x}) - D_k^l(\hat{\boldsymbol{x}})\|_1}{\text{mean}\left(\|D_k^l(\boldsymbol{x})\|_1\right)},$$

where the mean is computed over all dimensions, (D_k) are the discriminators, and L is the number
discriminators. The discriminators are trained to minimize the following hinge-loss adversarial los
$L_d(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{K}\sum_{k=1}^{K}\max(0, 1 - D_k(\boldsymbol{x})) + \max(0, 1 + D_k(\hat{\boldsymbol{x}}))$, where K is the number of discriminat
that the discriminator tend to overpower easily the decoder, we update its weight with a probabilit
24 kHz, and 0.5 at 48 kHz.

Encodec (2022)

Loss

RVQ Commitment Loss

VQ commitment loss. As mentioned in Section 3.2, we add a commitment loss l_w between th the encoder, and its quantized value, with no gradient being computed for the quantized value residual step $c \in \{1, \dots C\}$ (with C depeding on the bandwidth target for the current batch), no current residual and $q_c(\boldsymbol{z}_c)$ the nearest entry in the corresponding codebook, we define l_w as

$$l_w = \sum_{c=1}^{C} \|\boldsymbol{z}_c - q_c(\boldsymbol{z}_c)\|_2^2.$$

Overall, the generator is trained to optimize the following loss, summed over the batch,

$$L_G = \lambda_t \cdot \ell_t(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \lambda_f \cdot \ell_f(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \lambda_g \cdot \ell_g(\hat{\boldsymbol{x}}) + \lambda_{feat} \cdot \ell_{feat}(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \lambda_w \cdot \ell_w(w),$$

where λ_t, λ_f, λ_g, λ_{feat}, and λ_w the scalar coefficients to balance between the terms.

Encodec (2022)

Loss

Balancer

Balancer. We introduce a loss balancer in order to stabilize training, in particular the varying s gradients coming from the discriminators. We also find that the balancer makes it easier to reason different loss weights, independently of their scale. Let us take a number of losses $(\ell_i)_i$ that deper the output of the model $\hat{x}$. We define $g_i = \frac{\partial \ell_i}{\partial \hat{x}}$, and $\langle \|g_i\|_2 \rangle_\beta$ the exponential moving average of last training batches. Given a set of weights (λ_i) and a reference norm R, we define

$$\tilde{g}_i = R \frac{\lambda_i}{\sum_j \lambda_j} \cdot \frac{g_i}{\langle \|g_i\|_2 \rangle_\beta}.$$

We then backpropagate into the network $\sum_i \tilde{g}_i$, instead of the original $\sum_i \lambda_i g_i$. This changes the op problem but allows to make the λ_i interpretable irrespectively of the natural scale of each loss. If then each weight can be interpreted as the fraction of the model gradient that come from the cor loss. We take $R = 1$ and $\beta = 0.999$. All the generator losses from Eq. (4) fit into the balancer, the commitment loss, as it is not defined with respect to the output of the model.

Encodec (2022)

Evaluation

Table 1: MUSHRA scores for Opus, EVS, Lyra-v2, and EnCodec for various bandwidths streamable setting. Results are reported across different audio categories (clean speech, noisy s music), sampled at 24 kHz. We report mean scores and 95% confidence intervals. For EnCode report the average bandwidth after using the entropy coding described in Section 3.3.

Model	Bandwidth	Entropy Coded	Clean Speech	Noisy Speech	Music Set-1	Mus
Reference	-	-	95.5±1.6	93.9±1.8	93.2±2.5	97
Opus	6.0 kbps	-	30.1±2.8	19.1±5.9	20.6±5.8	17
Opus	12.0 kbps	-	76.5±2.3	61.9±2.1	77.8±3.2	65
EVS	9.6 kbps	-	84.4±2.5	80.0±2.4	89.9±2.3	87
Lyra-v2	3.0 kbps	-	53.1±1.9	52.0±4.7	69.3±3.3	42
Lyra-v2	6.0 kbps	-	66.2±2.9	59.9±3.3	75.7±2.6	48
EnCodec	1.5 kbps	0.9 kbps	49.2±2.4	41.3±3.6	68.2±2.2	66
EnCodec	3.0 kbps	1.9 kbps	67.0±1.5	62.5±2.3	89.6±3.1	87
EnCodec	6.0 kbps	4.1 kbps	83.1±2.7	69.4±2.3	92.9±1.8	91
EnCodec	12.0 kbps	8.9 kbps	90.6±2.6	80.1±2.5	91.8±2.5	92

Audio Codecs

🤖**Models:**

- ✅Encodec (2022)
- Mimi (2024)

Mimi (2024)

Architecture

- NEW Semantic Features Distillation

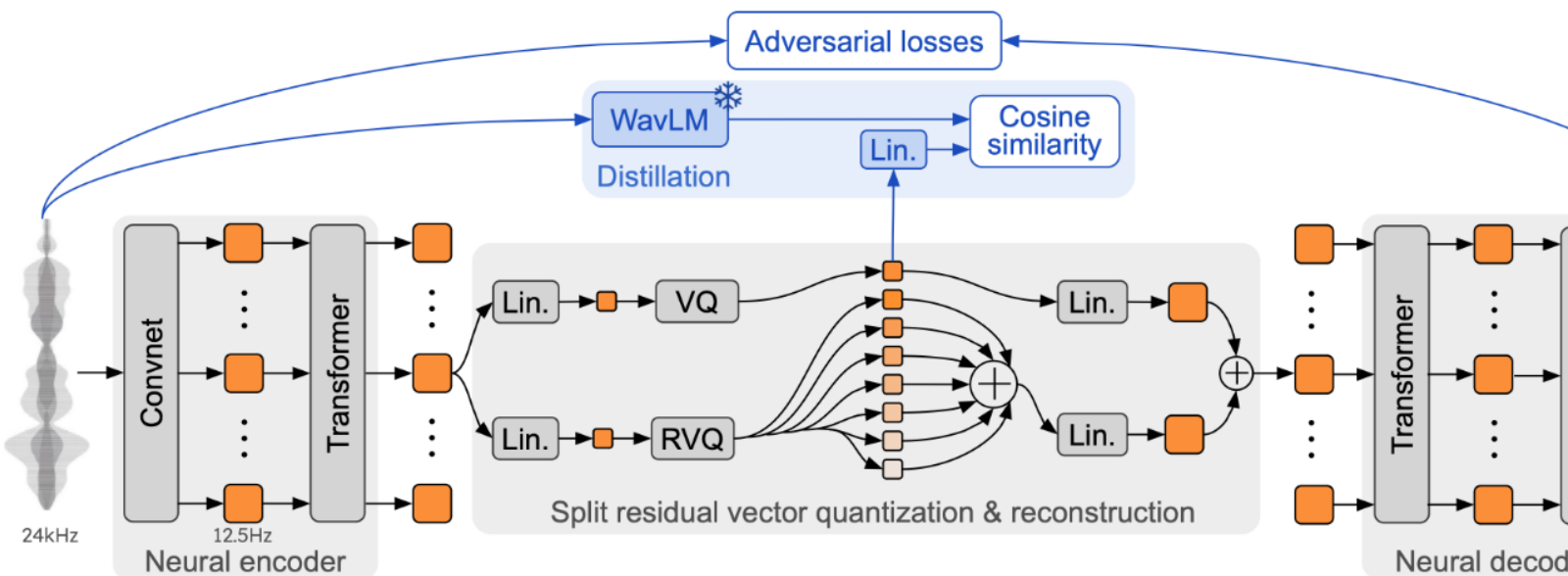


Figure 2: **Architecture and training of Mimi, our neural audio codec, split residual vector quantization**. During training (blue part, top), we di causal embeddings from WavLM (Chen et al., 2022) into a single vector quantiz produces semantic tokens, and is combined with separate acoustic tokens for recons

Mimi (2024)

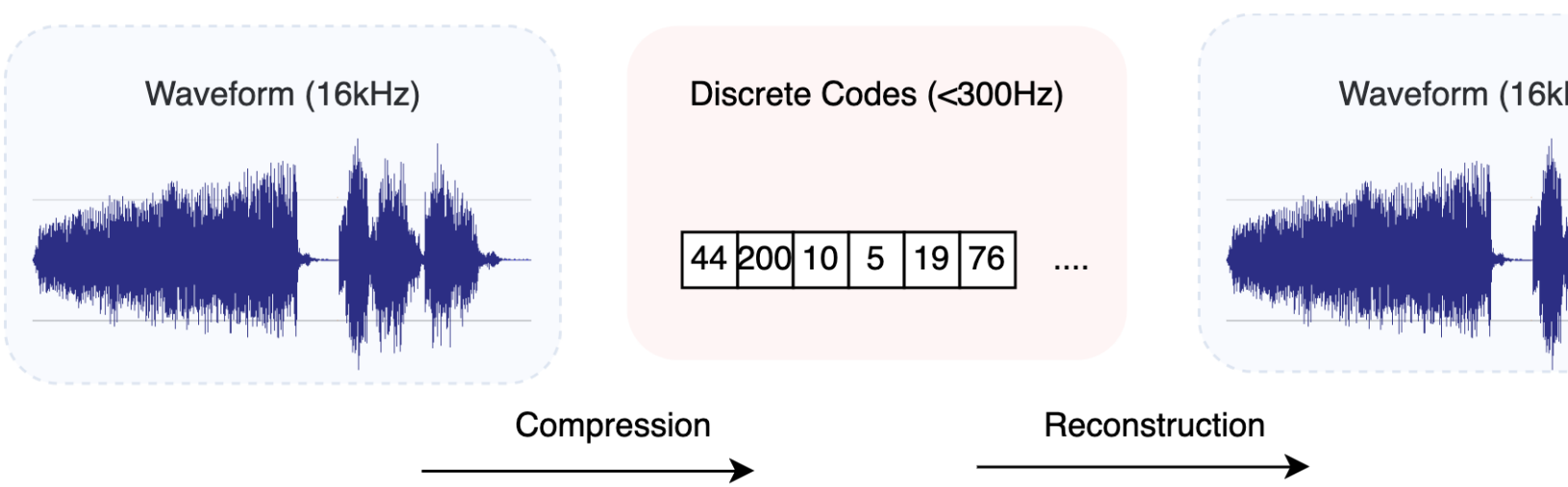
Metrics

Table 4: **Audio quality evaluation**. Objective and subjective (MUSHRA) eval
audio quality for baseline neural audio codecs—RVQGAN (Kumar et al., 2024)
tiCodec (Liu et al., 2024), and SpeechTokenizer (Zhang et al., 2024b)— and
important variants of Mimi. For a fair comparison with SemantiCodec and Sp
enizer, we also include a downsampled version of our codec in the MUSHRA stu
the audio sample rate and f_r the codec frame rate. Both Mimi codecs are trai
distillation, and either with the same combination of reconstruction and adversar
as Encodec (see Section 3.3) or adversarial losses only.

Model	f_s	f_r	bitrate	causal	ABX (↓)	VisQOL (↑)	MOSNet (↑)	MU
Ground Truth	24kHz	-	-	-	-	-	3.08	
RVQGAN	24kHz	75Hz	1.5kbps		-	1.74	2.74	
SemantiCodec	16kHz	50Hz	1.3kbps		42.2%	2.43	3.12	
SpeechTokenizer	16kHz	50Hz	1.5kbps		**3.3%**	1.53	2.67	
SpeechTokenizer	16kHz	50Hz	4.0kbps		**3.3%**	3.07	3.10	
Mimi, adv. loss only	24kHz	12.5Hz	1.1kbps	✓	8.7%	1.84	3.10	
Same, downsampled at 16kHz	16kHz	12.5Hz	1.1kbps	✓	-	-	-	
Mimi, non adv. only	24kHz	12.5Hz	1.1kbps	✓	8.1%	2.82	2.89	

Audio Codecs

- ✅Encodec
- ✅Mimi



Audio Codecs

Feature	Mimi	Encodec (Meta
Input audio sample rate	16 kHz	24,48 kHz
Codec sample rate	12.5 Hz	150-2400 Hz
Varying bandwidth	❌	✅
Semantic distillation	✅	❌

Coversational LLMs

- ☑ Audio Understanding
- ☑ Audio Generation
- ☑ Emotional
- ☑ Interruptions handling
- ☑ Instructions following (world model)

Coversational LLMs

Live demo of GPT4-o voice variation

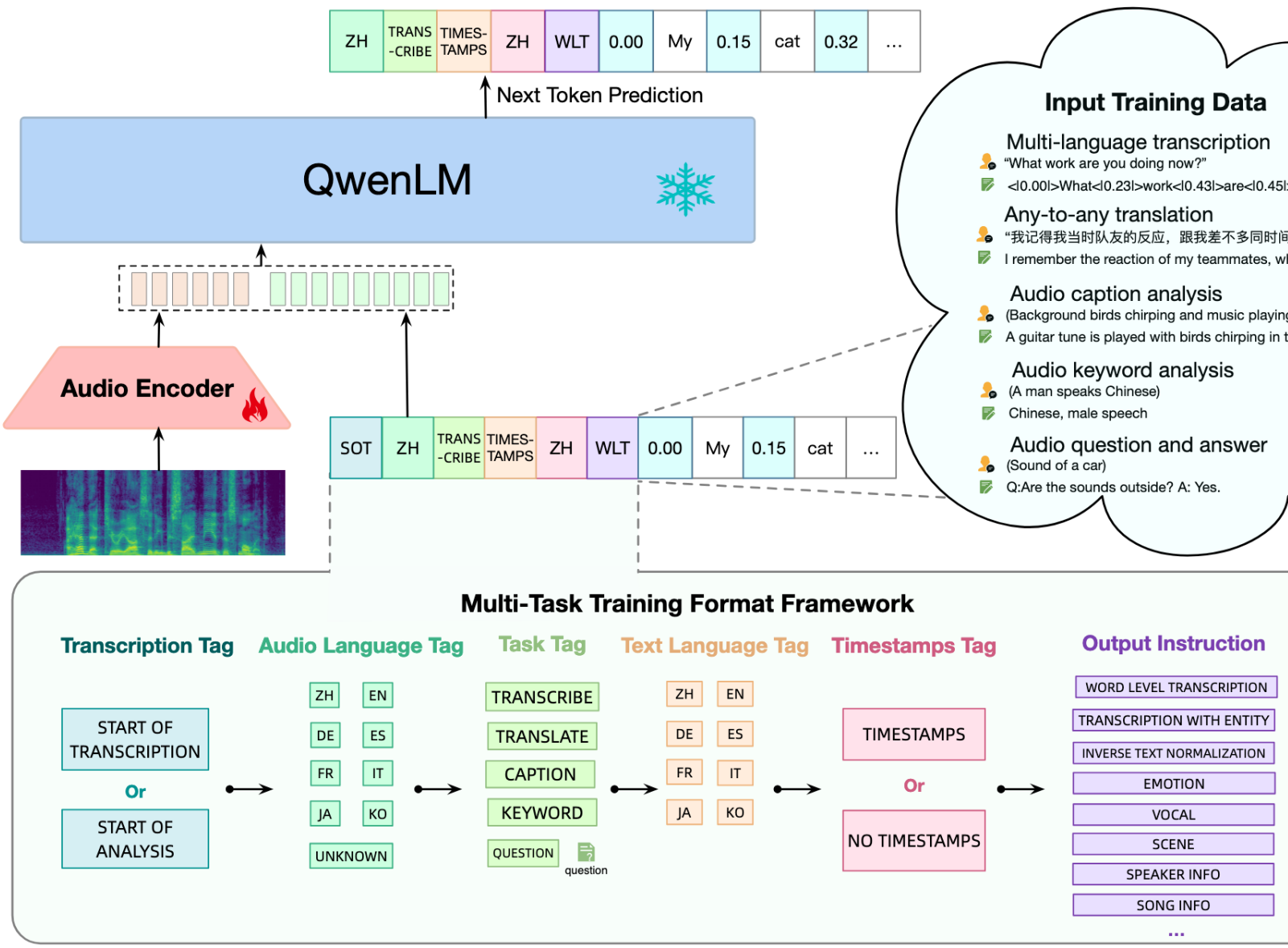
Audio LLMs

- Qwen2.5-Audio 👈
- Qwen2.5-Omni
- Moshi

Qwen2.5-Audio.

Architecture.

- Whisper-Initialized Audio Encoder
- Qwen2.5-7B LLM Backbone


ZH
TRANS -CRIBE
TIMES- TAMPS
ZH
WLT
0.00
My
0.15
cat
0.32
...
Next Token Prediction
QwenLM
Audio Encoder
SOT
ZH
TRANS -CRIBE
TIMES- TAMPS
ZH
WLT
0.00
My
0.15
cat
...
Input Training Data
Multi-language transcription
"What work are you doing now?"
<|0.00|>What<|0.23|>work<|0.43|>are<|0.45|
Any-to-any translation
"我记得我当时队友的反应，跟我差不多同时间
I remember the reaction of my teammates, wh
Audio caption analysis
(Background birds chirping and music playing
A guitar tune is played with birds chirping in t
Audio keyword analysis
(A man speaks Chinese)
Chinese, male speech
Audio question and answer
(Sound of a car)
Q:Are the sounds outside? A: Yes.
Multi-Task Training Format Framework
Transcription Tag
START OF TRANSCRIPTION
Or
START OF ANALYSIS
Audio Language Tag
ZH
EN
DE
ES
FR
IT
JA
KO
UNKNOWN
Task Tag
TRANSCRIBE
TRANSLATE
CAPTION
KEYWORD
QUESTION
question
Text Language Tag
ZH
EN
DE
ES
FR
IT
JA
KO
Timestamps Tag
TIMESTAMPS
Or
NO TIMESTAMPS
Output Instruction
WORD LEVEL TRANSCRIPTION
TRANSCRIPTION WITH ENTITY
INVERSE TEXT NORMALIZATION
EMOTION
VOCAL
SCENE
SPEAKER INFO
SONG INFO
...

Qwen2.5-Audio.

Training.

2 stages training:

1. **Multitask pretraining**:

    🔥Audio Encoder. ❄️LLM.

2. **SFT**:

    ❄️Audio Encoder. 🔥LLM.

Qwen2.5-Audio.

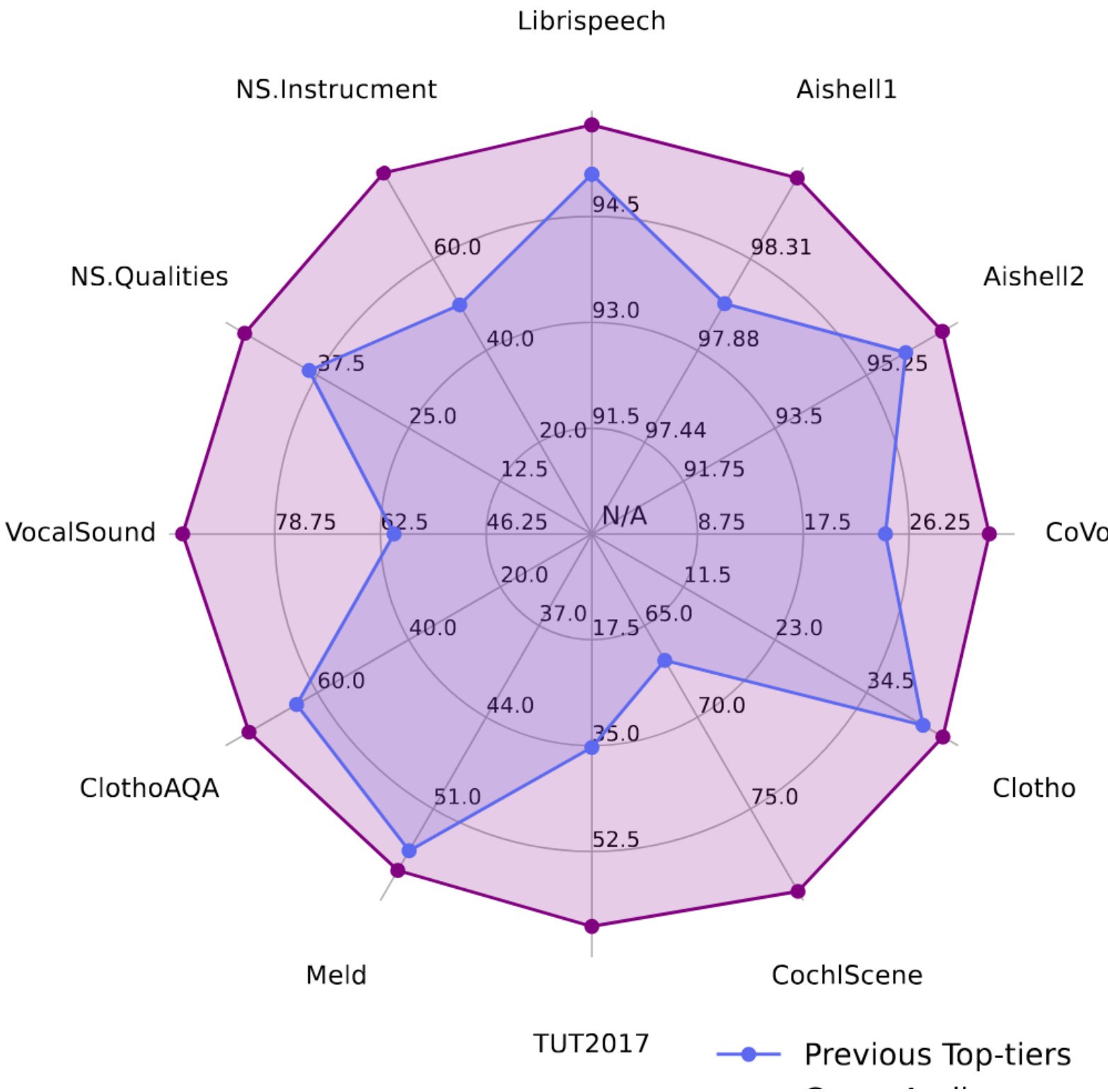
Tasks.

Types	Task	Description
Speech	ASR	Automatic speech recognition (multiple languages)
	S2TT	Speech-to-text translation
	OSR	Overlapped speech recognition
	Dialect ASR	Automatic dialect speech recognition
	SRWT	English speech recognition with word-level timestamps
		Mandarin speech recognition with word-level timestamps
	DID	Dialect identification
	LID	Spoken language identification
	SGC	Speaker gender recognition (biologically)
	ER	Emotion recognition
	SV	Speaker verification
	SD	Speaker diarization
	SER	Speech entity recognition
	KS	Keyword spotting
	IC	Intent classification
	SF	Slot filling
	SAP	Speaker age prediction
	VSC	Vocal sound classification
Sound	AAC	Automatic audio caption
	SEC	Sound event classification
	ASC	Acoustic scene classification
	SED	Sound event detection with timestamps
	AQA	Audio question answering
Music&Song	SID	Singer identification
	SMER	Singer and music emotion recognition
	MC	Music caption
	MIC	Music instruments classification
	MNA	Music note analysis such as pitch, velocity
	MGR	Music genre recognition

Qwen2.5-Audio.

Results.

- ⚠️No Text Modality Metrics were reported.

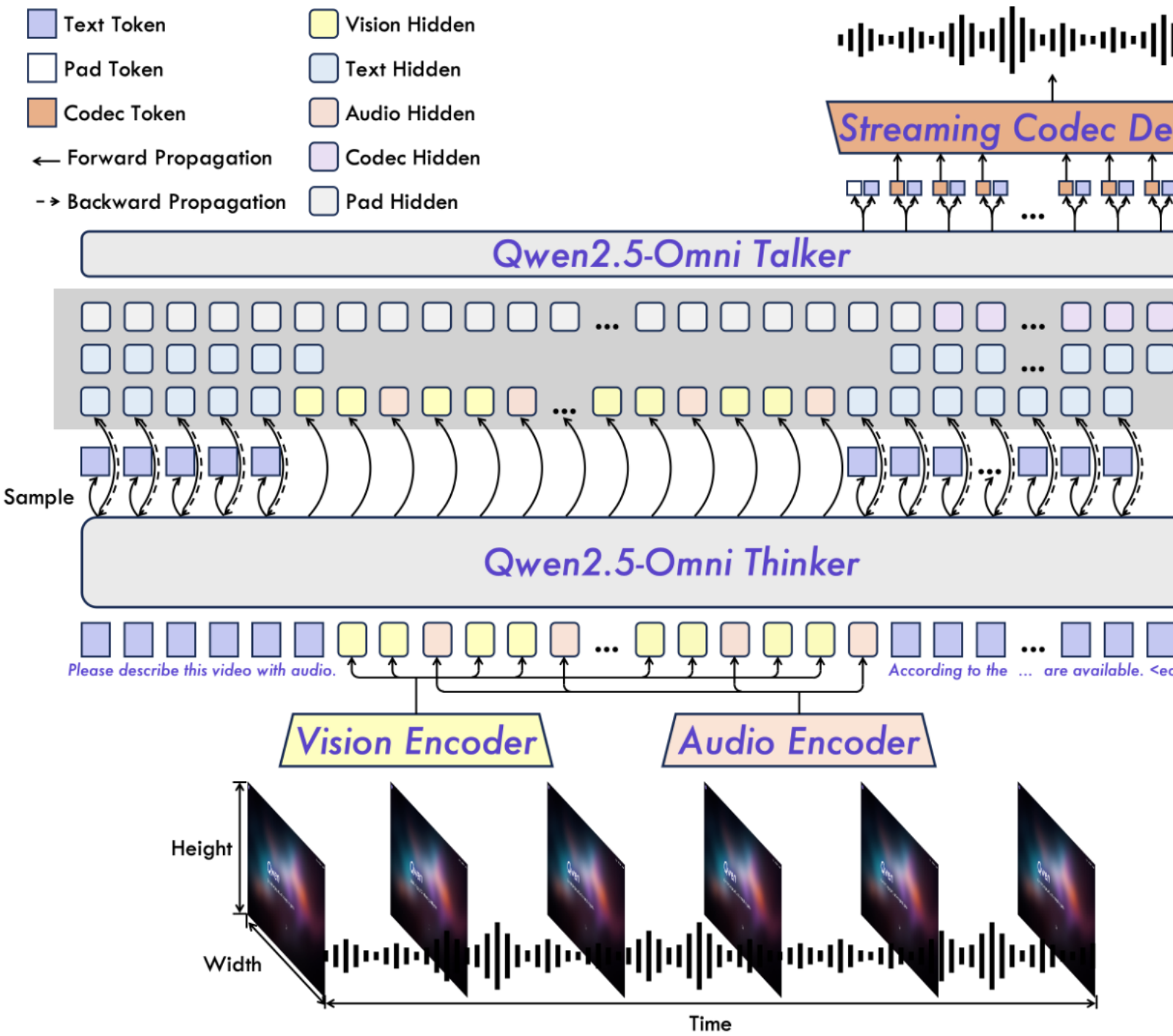

Librispeech
Aishell1
Aishell2
CoVo
Clotho
CochlScene
TUT2017
Meld
ClothoAQA
VocalSound
NS.Qualities
NS.Instrucment
94.5
60.0
98.31
93.0
40.0
97.88
37.5
95.25
25.0
91.5
97.44
93.5
20.0
12.5
91.75
N/A
78.75
62.5
46.25
8.75
17.5
26.25
20.0
11.5
37.0
65.0
17.5
23.0
40.0
60.0
34.5
44.0
70.0
35.0
51.0
75.0
52.5
Previous Top-tiers

Audio LLMs

- Qwen2.5-Audio
- Qwen2.5-Omni 👈
- Moshi

Qwen2.5-Omni.

Architecture.


Text Token
Pad Token
Codec Token
Forward Propagation
Backward Propagation
Vision Hidden
Text Hidden
Audio Hidden
Codec Hidden
Pad Hidden
Streaming Codec De
Qwen2.5-Omni Talker
Sample
Qwen2.5-Omni Thinker
Please describe this video with audio.
According to the ... are available. <ea
Vision Encoder
Audio Encoder
Height
Width
Time

Qwen2.5-Omni.

Thinker Training.

3 training stages:

1. **Encoders pretraining**:

   🔥Audio Encoder. 🔥Image Encoder. ❄️LLM.

2. **Finetuning**:

   🔥Audio Encoder. 🔥Image Encoder. 🔥LLM.

3. **Long Context Finetuning - 32k**:

   🔥Audio Encoder. 🔥Image Encoder. 🔥LLM.

Qwen2.5-Omni.

Talker Training.

3 training stages:

1. **Pretraining**:

   ❄️Thinker. 🔥Talker.

2. **DPO**:

   ❄️Thinker. 🔥Talker.

3. **SFT Instruciton-tuning**:

   ❄️Thinker. 🔥Talker.

Qwen2.5-Omni.

Results.

Text Benchmarks.

Datasets	Qwen2-7B (text)	Qwen2-Audio	Qwen2.5-Om
MMLU*	69.3	33.2	65.6
CEval*	78.4	38.6	61.1
IFEval*	53.3	15.6	41.7
GSM8K*	82.3	18.4	85.4
Math23K*	92.3	23.0	87.1
Math401*	75.5	20.4	62.2

Qwen2.5-Omni.

Results.

Audio Understanding Benchmarks.

Table 8: Multimodality → Text performance of State-of-the-art and Qwen2.5-Omni

Datasets	Model	Performance
	Multimodal Understanding	
OmniBench *Speech* | *Sound Event* | *Music* | *Avg*	Gemini-1.5-Pro (Team et al., 2024)	42.67% | 42.26% | 46.23%
	MIO-Instruct (Wang et al., 2024g) (7B)	36.96% | 33.58% | 11.32%
	AnyGPT (7B) (Zhan et al., 2024)	17.77% | 20.75% | 13.21%
	video-SALMONN (13B) (Sun et al., 2024)	34.11% | 31.70% | **56.60%**
	UnifiedIO2-xlarge (3.2B) (Lu et al., 2024a)	39.56% | 36.98% | 29.25%
	UnifiedIO2-xxlarge (6.8B) (Lu et al., 2024a)	34.24% | 36.98% | 24.53%
	MiniCPM-o (Yao et al., 2024)	- | - | - | 40.5%
	Baichuan-Omni-1.5 (Li et al., 2025)	- | - | - | 42.9%
	Qwen2.5-Omni-7B	**55.25%** | **60.00%** | 52.83%

Qwen2.5-Omni.

Results.

Zero-shot Speech Generation Benchmarks.

Table 9: Zero-Shot Speech Generation

Datasets	Model	Performance
	Content Consistency	
SEED *test-zh* | *test-en* | *test-hard*	Seed-TTS$_{\text{ICL}}$ (Anastassiou et al., 2024)	1.11 | 2.24 | 7.
	Seed-TTS$_{\text{RL}}$ (Anastassiou et al., 2024)	**1.00** | 1.94 | **6.**
	MaskGCT (Wang et al., 2024e)	2.27 | 2.62 | 1(
	E2 TTS (Eskimez et al., 2024)	1.97 | 2.19 | -
	F5-TTS (Chen et al., 2024c)	1.56 | **1.83** | 8.
	CosyVoice 2 (Du et al., 2024)	1.45 | 2.57 | 6.
	CosyVoice 2-S (Du et al., 2024)	1.45 | 2.38 | 8.
	Qwen2.5-Omni-7B$_{\text{ICL}}$	1.70 | 2.72 | 7.
	Qwen2.5-Omni-7B$_{\text{RL}}$	1.42 | 2.33 | 6.
	Speaker Similarity	
SEED *test-zh* | *test-en* | *test-hard*	Seed-TTS$_{\text{ICL}}$ (Anastassiou et al., 2024)	0.796 | 0.762 |
	Seed-TTS$_{\text{RL}}$ (Anastassiou et al., 2024)	**0.801** | **0.766** |
	MaskGCT (Wang et al., 2024e)	0.774 | 0.714 |
	E2 TTS (Eskimez et al., 2024)	0.730 | 0.710 |
	F5-TTS (Chen et al., 2024c)	0.741 | 0.647 |
	CosyVoice 2 (Du et al., 2024)	0.748 | 0.652 |
	CosyVoice 2-S (Du et al., 2024)	0.753 | 0.654 |
	Qwen2.5-Omni-7B$_{\text{ICL}}$	0.752 | 0.632 |
	Qwen2.5-Omni-7B$_{\text{RL}}$	0.754 | 0.641 |

Qwen2.5-Omni.

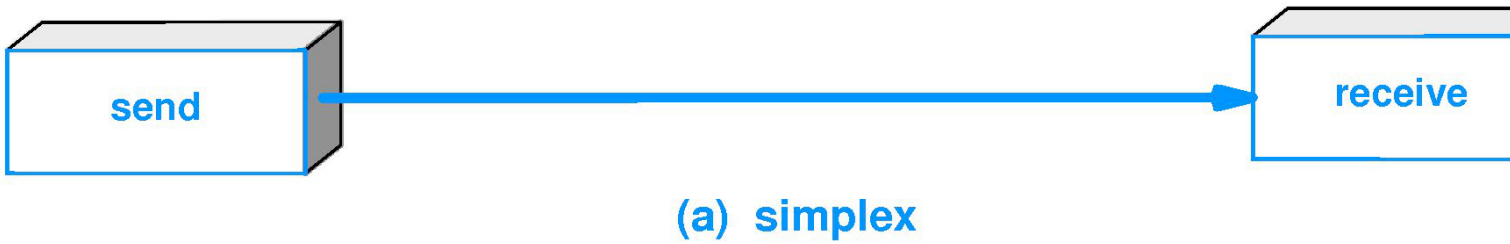
Questions.

- Qwen-TTS-Tokenizer (**issue**) ❓
- Training Tasks and Data Details ❓
- Why talker have to generate text tokens ❓ W
  about tasks interference ❓

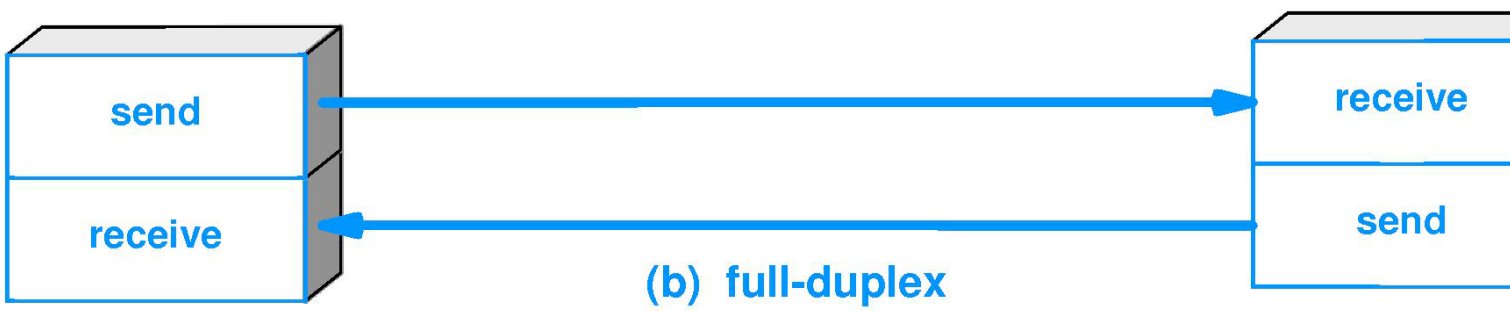
Audio LLMs

- Qwen2.5-Audio
- Qwen2.5-Omni
- Moshi 👈

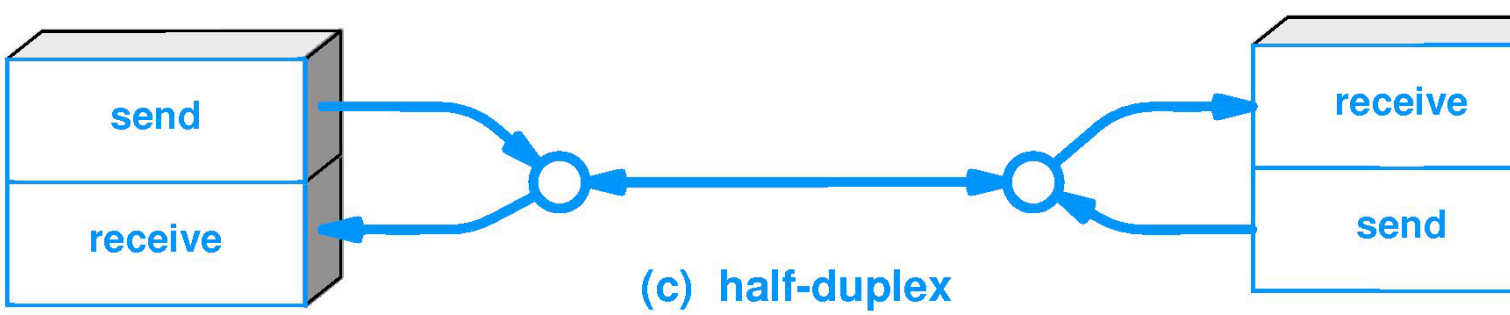
Simplex / Duplex / Half-duplex



(a) simplex



(b) full-duplex



(c) half-duplex

Moshi

Architecture.

- Pretrained LLM: Helium-7B
- Full-duplex

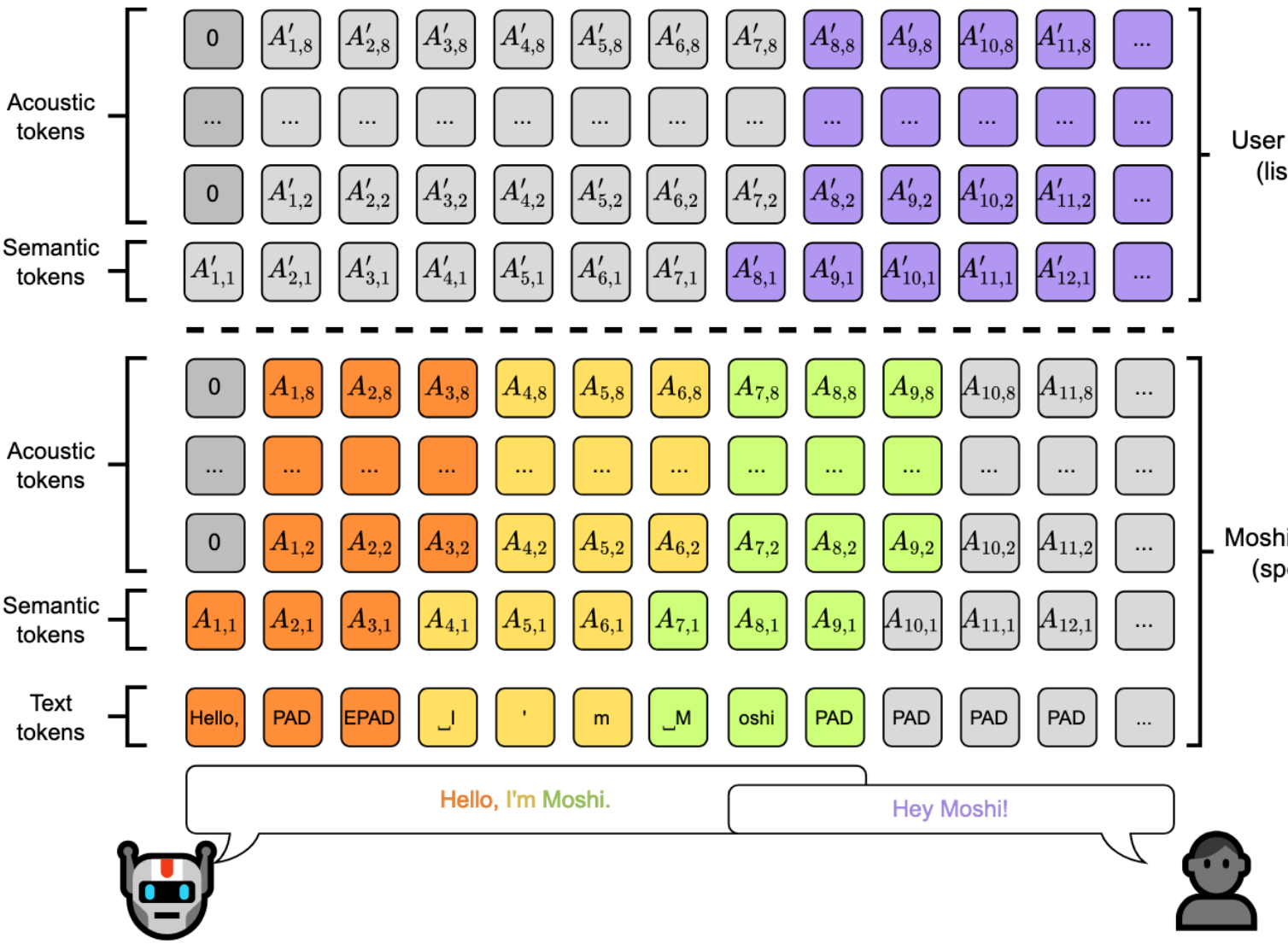


Figure 4: **Representation of the joint sequence modeled by Moshi.** Each represents the tokens for a given step in the joint sequence ($V_{s,k}$) described in Ec with an acoustic delay $\tau = 1$, e.g. the input of the Temporal Transformer for t Tokens are predicted from bottom to top in the Depth Transformer. At inferen tokens under the dashed line (corresponding to Moshi) are sampled, while those a fed from the user. This design allows for our model to handle overlapping speech

Moshi

Training.

3 training stages:

1. 🎤🗣️📚Audio Pretraining

   Also text pretraing to prevent catastrophic forgetting

2. 🎤🗣️↔️📚Full-Duplex Training (Synth Data)
3. 🎤🗣️↔️🫧Clean Dialogue Dataset SFT

Moshi

Evaluation.

Table 7: **Performance of audio and text language modeling**. We report a based on scoring with negative log-likelihood, normalized by sequence length. evaluated in a 5-shot setting. Reusing the terminology of Nguyen et al. (2024), ∅ r unsupported modalities while - represents unreported numbers.

	Audio metrics			
Model	sWUGGY	sBLIMP	sTopic-StoryCloze	sStoryCloze
Audio only - Cold Start				
GSLM (Lakhotia et al., 2021)	64.8	54.2	66.6	53.3
AudioLM (Borsos et al., 2022)	71.5	**64.7**	-	-
TWIST (Hassid et al., 2023)	72.2	56.5	-	-
Moshi	**74.8**	59.9	80.9	56.9
Audio only - Warm Start				
TWIST (Hassid et al., 2023)	74.5	59.2	76.4	55.4
VoxtLM (Maiti et al., 2023)	62.9	53.9	-	-
Spirit-LM (Nguyen et al., 2024)	69.5	58.0	72.9	54.8
Moshi	74.3	58.9	81.8	58.7
Text and audio - Warm Start				
VoxtLM (Maiti et al., 2023)	66.1	57.1	-	-
Spirit-LM (Nguyen et al., 2024)	69.0	58.3	82.9	61.0
Moshi after single-stream pretraining	72.6	58.8	83.0	60.8
Moshi after multi-stream instruct	63.0	55.2	**83.6**	**62.7**
Moshi after multi-stream instruct, synthetic voice	60.9	54.6	82.5	60.9

Comparison

	Qwen2.5-Audio	Qwen2.5-Omni	Moshi (C
Audio-In	✅	✅	✅
Audio-Out		✅	✅
Image-In		✅	
Full-Duplex			✅

Materials

- Moshi overview SpeechInfo [1/2], [2/2]
- **Sesame Conversational Voice**