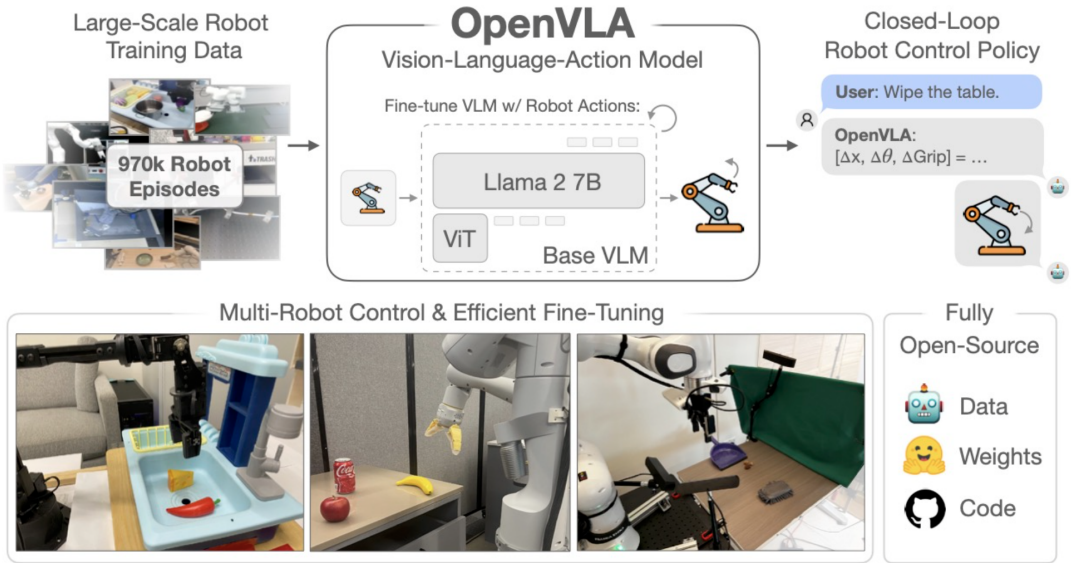
Vision-Language-Action Models

RT1, RT2, OpenVLA

A. Buynitsky

Jan 26, 2025



Outline

Outline

FiLM Layers

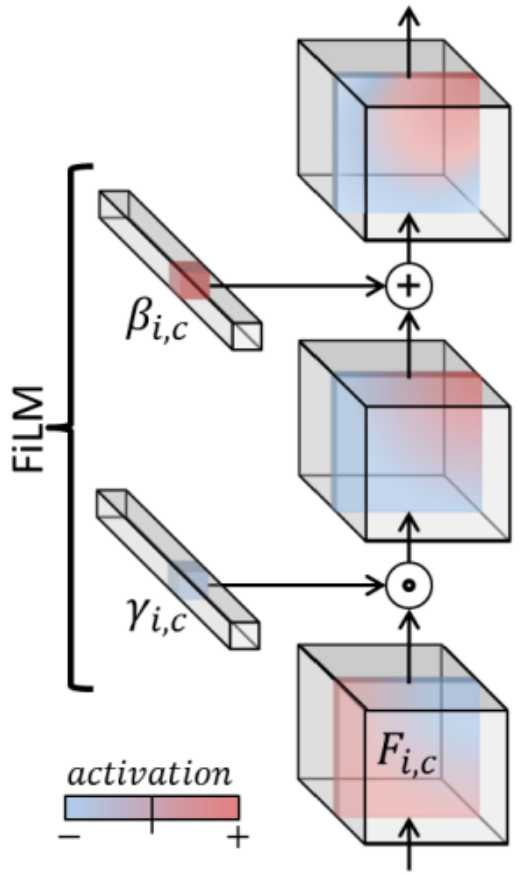
FiLM adaptively influence output of neural network by applying a affine (linear) transformation to intermediate layers.

FiLM learns functions f and h based on external input x_i (i.e image) in a batch

$$\gamma_{i,c} = f_c(x_i) \quad \beta_{i,c} = h_c(x_i)$$

$$\text{FiLM}(F_{i,c} \mid \gamma_{i,c}, \beta_{i,c}) = \gamma_{i,c} F_{i,c} + \beta_{i,c}$$

$F_{i,c}$ is the c^{th} feature of the i^{th} sample in the batch



FiLM Layers

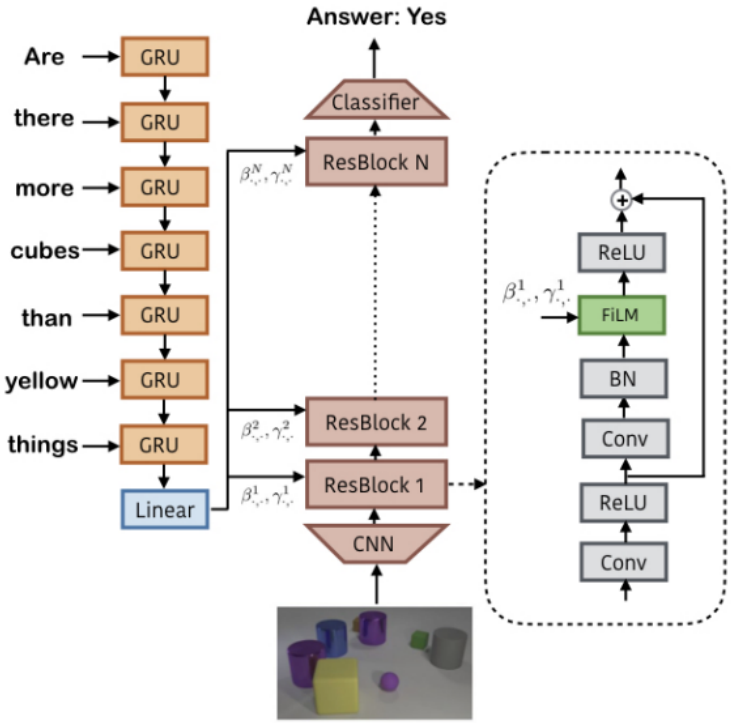
FiLM adaptively influence output of neural network by applying a affine (linear) transformation to intermediate layers.

FiLM learns functions f and h based on external input x_i (i.e image) in a batch

$$\gamma_{i,c} = f_c(x_i) \quad \beta_{i,c} = h_c(x_i)$$

$$\text{FiLM}(F_{i,c} \mid \gamma_{i,c}, \beta_{i,c}) = \gamma_{i,c} F_{i,c} + \beta_{i,c}$$

$F_{i,c}$ is the c^{th} feature of the i^{th} sample in the batch



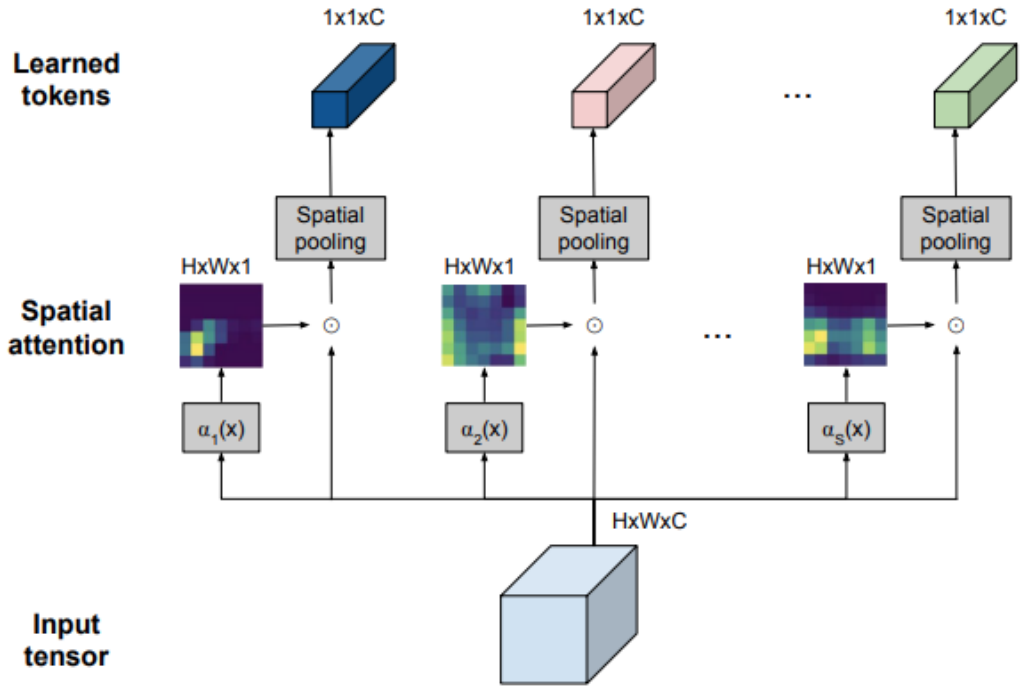
TokenLearner

Goal: Generate $[z_i]_{i=1}^{S} \in \mathbb{R}^{S \times C}$ from $x \in \mathbb{R}^{H \times W \times C}$ by learning S functions A_i to adaptively select informative combo of pixels in x_t denoted as:

$$z_i = A_i(x)$$

Implement with weight map $\alpha_i(x)$ and spatial global average pooling $\rho(x)$:

$$z_i = A_i(x) = \rho(x \odot \gamma(\alpha_i(x)))$$

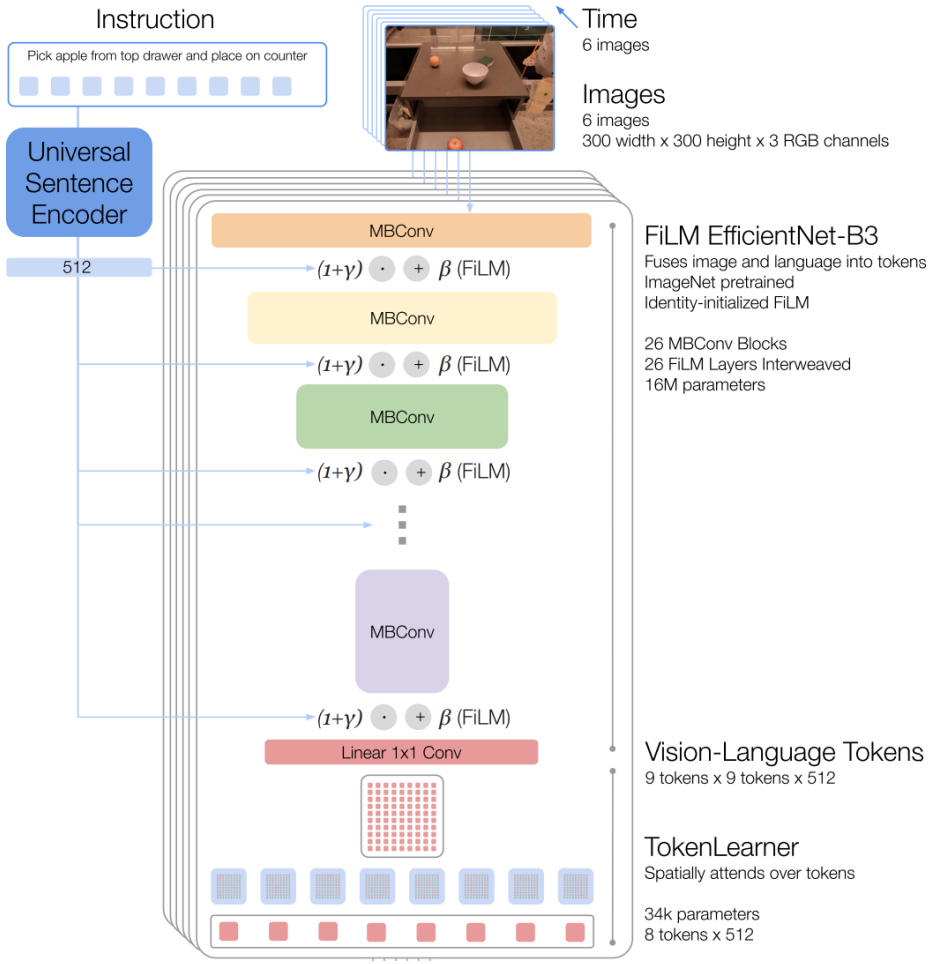


RT1 Architecture (Part 1)

- **Unversal Sentence Encoder:** Encoder block of Transformer

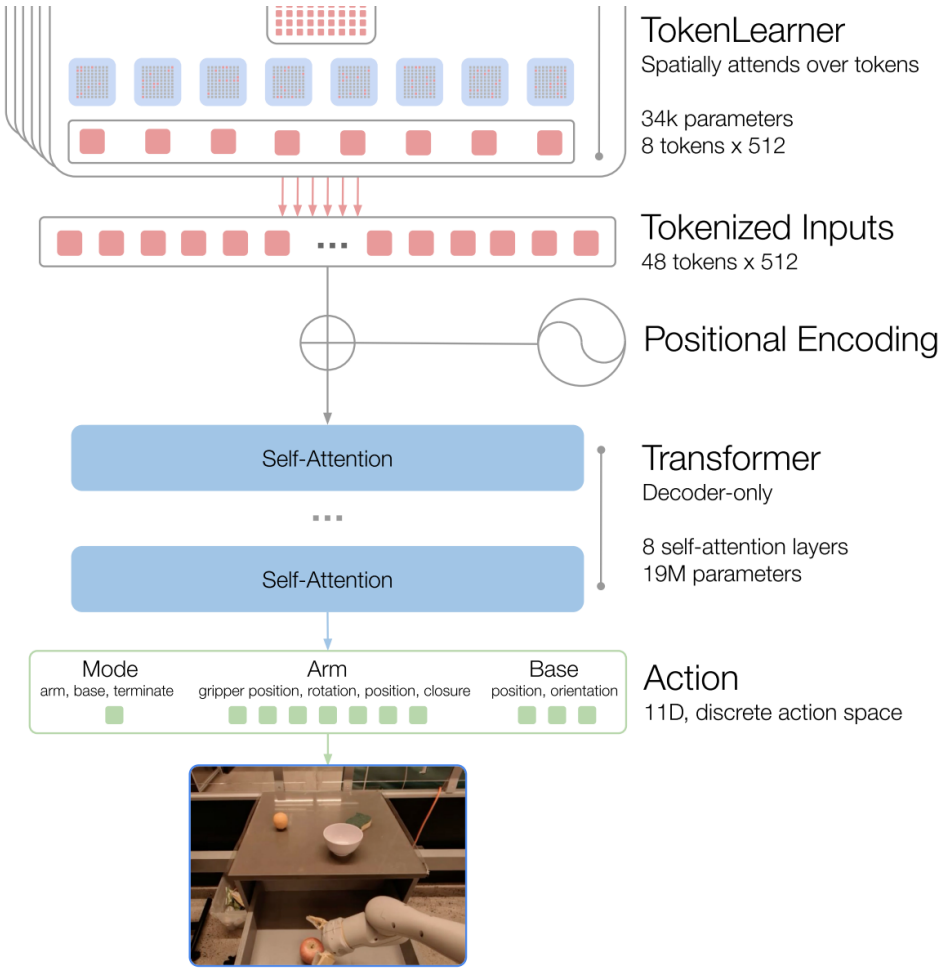
- **FiLM Layers:** Conditions EfficientNet on text
  TokenLearner: Donwsample 81 to 8 tokens per image

- **Transformer:** Apply transformer to FiLM output



RT1 Architecture (Part 2)

- **History:** 6-image history for total of 48 tokens
- **Transformer:** decoder-only arch with 8 self-attn layers
- **Action Tokenization:** Discretize continuous actions to 256 bins:
  - **Gripper Actions:** $x, y, z, \rho, \phi, \theta$, opening of gripper
  - **Base Actions:** x, y head angle
  - **mode:** control arm, control base or terminate



Outline

Converting VLMs to VLAs

Goal: Associate actions from model's existing tokenization for:

$$\text{terminate}\ \Delta\text{pos}_x\ \Delta\text{pos}_y\ \Delta\text{pos}_z\ \Delta\text{rot}_x\ \Delta\text{rot}_y\ \Delta\text{rot}_z\ \text{gripper\_extension}$$

Possible instantiation: "1 128 91 241 5 101 127"

PaLI-X Tokenization: Integers up to 1000 each have unique token, so associate action bins to token corresponding to integer
PaLM-E Tokenization: Overwrite the 256 least frequently used tokens to represent action vocabulary.
Co-Fine-Tuning: Train with both robotics data "Q: what action should robot take to [task instruction]? A:" and original web data.

RT2 Architecture

Prefix-decoder-only LLMs:
LLM is auto-regressive: condition model on prompt (prefix $w_{1:n}$) consisting of token embeddings $w_i \in \mathcal{X} \subset \mathbf{R}^k$:

$$p(w_{n+1:L} \mid w_{1:n}) = \prod_{l=n+1}^{L} p_{\text{LM}}(w_l \mid w_{1:l-1})$$

Train end-to-end embeddings $\gamma : \mathcal{W} \to \mathcal{X}$:

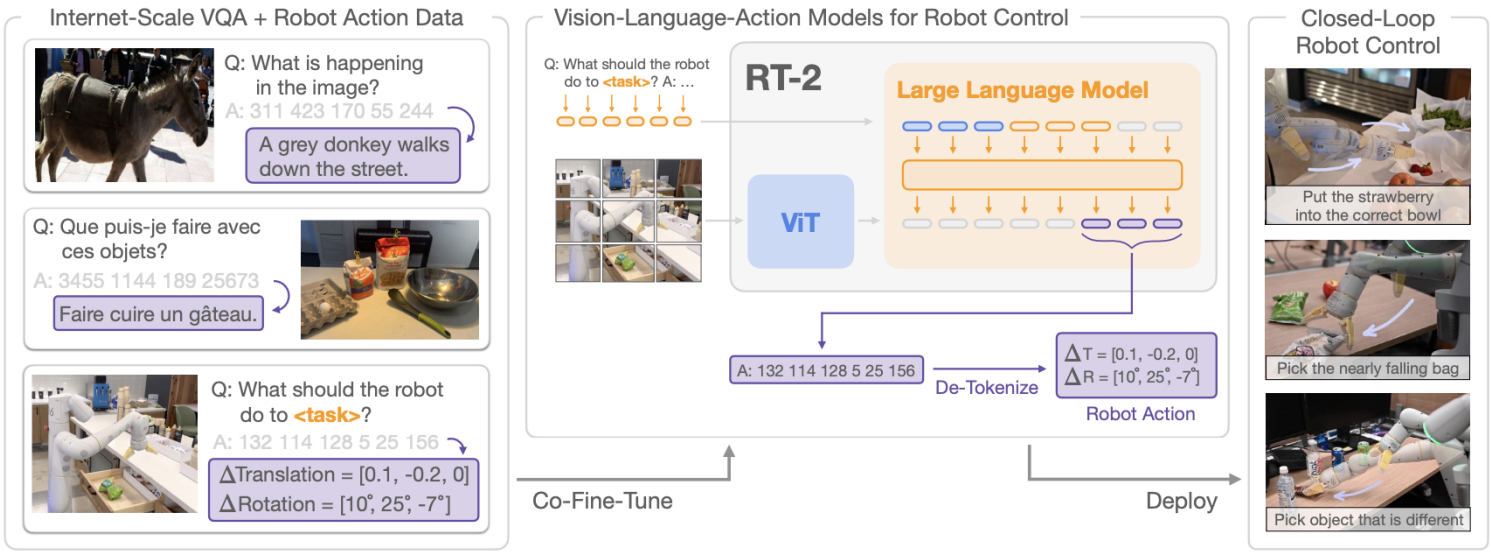
$$x_i = \gamma(w_i),$$

Adding Images:
ViT maps image I to tokens $\tilde{x}_{1:m} = \tilde{\phi}_{\text{ViT}}(I) \in \mathbf{R}^{m \times \tilde{k}}$
Project $\tilde{x}_{1:m}$ to embedding space via affine transformation $\psi : \mathbf{R}^{\tilde{k}} \to \mathbf{R}^k$

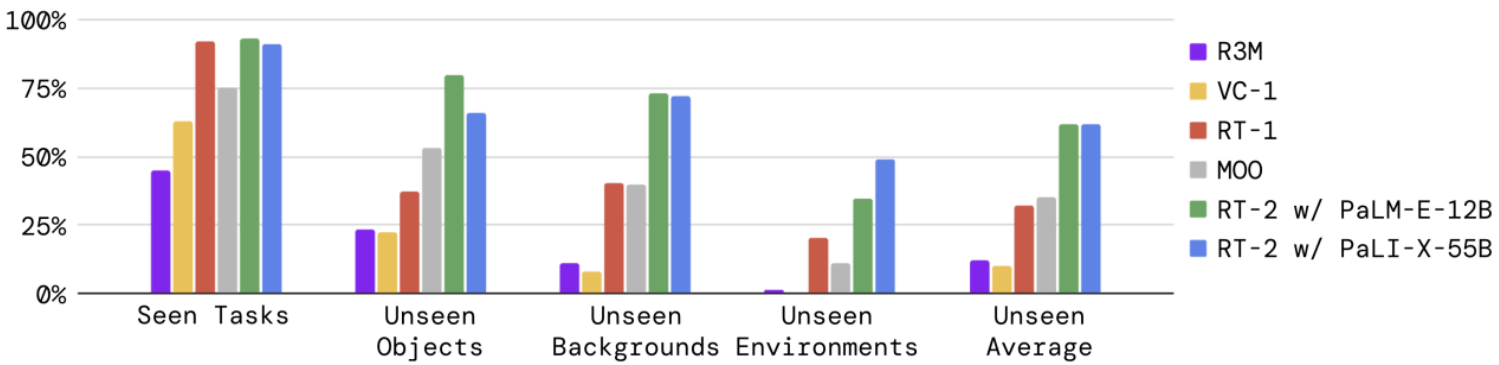
Robot State: (Joint angles, gripper state, etc.)
Project $s \in \mathbf{R}^S$ to embedding space via affine transformation $\psi : \mathbf{R}^S \to \mathbf{R}^k$

RT2 Architecture and Results

Complete Architecture:



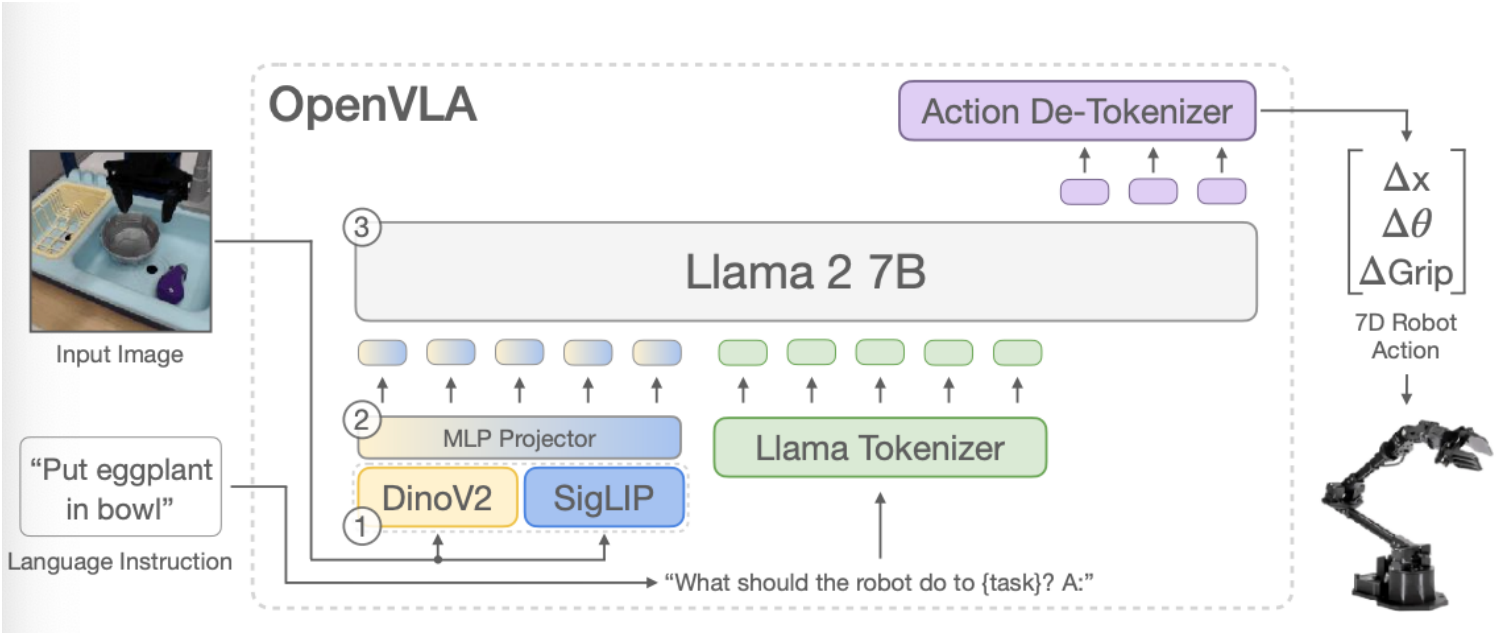
Results:



Outline

OpenVLA Architecture

Complete Architecture:



Vision Encoder:
Concatenate embeddings from SigLip and DinoV2 channelwise

Projection Layer
2-layer MLP projecting to embedding dimension of llama (512)

LLM Backbone:
Llama-2 7B

Data and Tokenization Details

Tokenizer

- LLama tokenizer reserves 100 toeksn for fine-tuning.
- Chose to follow RT2 tokenization. Discretize each dim of robot actions seperately into one of 256 bins.
- Replace 256 least frequent tokens with action tokens.

Training Data

- OpenX dataset (70 robot embeddings w/ ¿ 2M trajectories)
- Restrict datasets to contain only 1 manipulator with 3rd pov camera
- weight down / remove less diverse datasets, up-weight datasets with larger task and scene diversity

Training Details

- Decrease image resolution from 384×384 to 224×224 for $3\times$ training speedup
- Train until accuracy passes 95% (27 epochs using fixed lr of 2e-5)
- finetune vision encoder weights for better spatial understanding
- Train on 64 A100 GPUs for 14 days using batch size of 2048
- requires 15GB of GPU memory when loading in bfloat16

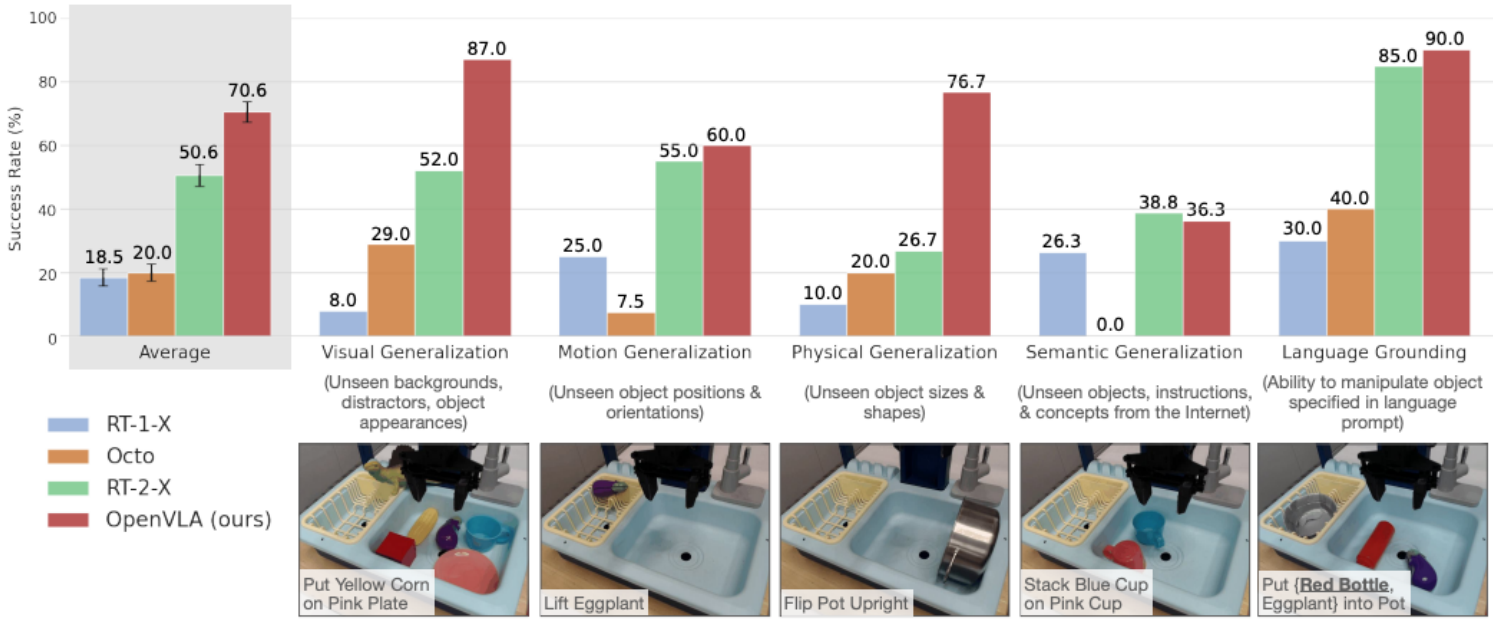
Fine-Tuning OpenVLA

- **full finetune:** updats all weights during training
- **last layer only:** finetunes only last layer of transformer backbone and embedding matrix
- **sandwich** finetunes vision encoder, embedding matrix and last layer
- **LoRA** applied to all layers of the model using varying rank $r \in 32, 64$

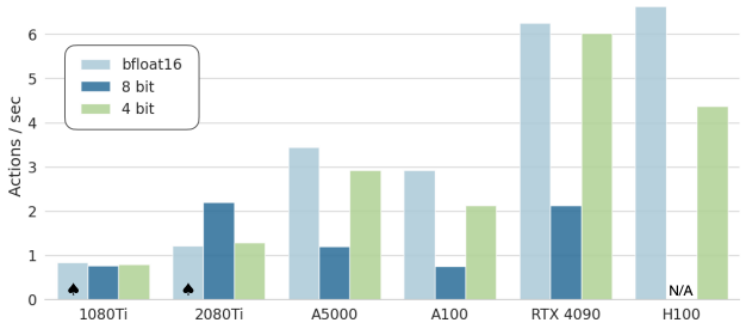
Strategy	Success Rate	Train Params ($\times 10^6$)	VRAM (batch 16)
Full FT	**69.7 ± 7.2 %**	7,188.1	163.3 GB*
Last layer only	30.3 ± 6.1 %	465.1	51.4 GB
Frozen vision	47.0 ± 6.9 %	6,760.4	156.2 GB*
Sandwich	62.1 ± 7.9 %	914.2	64.0 GB
LoRA, rank=32	**68.2 ± 7.5%**	**97.6**	**59.7 GB**
rank=64	**68.2 ± 7.8%**	195.2	60.5 GB

Results and Quantization

Overall Results:



Inference Speed:



Quantization Results:

Precision	Bridge Success	VRAM
bfloat16	$71.3 \pm 4.8\%$	16.8 GB
int8	$58.1 \pm 5.1\%$	10.2 GB
int4	$71.9 \pm 4.7\%$	7.0 GB

Outline

Aloha

TODO

Thank you!

Have a great rest of your Day!!!