# Review Paper - Octo: An Open-Source Generalist Robot Policy

Nikhil Rane
*Institute for Data Science in Mechanical Engineering*
*RWTH Aachen*
nikhil.rane@rwth-aachen.de
31st January, 2025

*Abstract*—**The development of Generalist Robot Policies (GRPs) represents a significant advancement in robotic control, enabling unified models to manage diverse tasks, sensor setups, and robotic platforms. Octo, an open-source, transformer-based GRP, addresses the limitations of existing models like RT-X, RoboCat, and GNM, which are often constrained by restrictive inputs, closed-source frameworks, and limited adaptability. Octo's innovative architecture integrates language, vision, and proprioceptive data into a unified tokenized format, leveraging pretrained encoders and diffusion-based action heads for generating smooth, multi-modal actions. Octo is trained on the Octo Data Mix, a curated subset of 25 datasets from the Open X-Embodiment dataset, comprising 800k high-quality trajectories. Octo demonstrates exceptional performance in zero-shot generalization and fine-tuning tasks. [6]**

*Index Terms*—**Multiembodiment Model, Robotics, Open X-Embodiment Dataset, Octo, Generalist Robot Policy, Diffusion Models in Robotics**

## I. Introduction

Robots have traditionally been designed to perform specific tasks within controlled environments, limiting their adaptability to new challenges. This specialization often requires developing separate control policies for each task and robot, leading to fragmented and resource intensive solutions.

The emergence of Generalist Robot Policies (GRPs) offers a promising alternative by enabling a single model to handle a wide range of tasks across various robots and sensor configurations. GRPs aim to streamline robotic learning by allowing models pretrained on diverse datasets to be fine-tuned with minimal data for new applications, enhancing efficiency and scalability.

Robotic foundation models are models designed to perform a wide variety of robotic tasks, often across a range of different robots. These models serve as a foundation or starting point for developing more specialized robotic systems and applications.

Recent efforts in GRPs include models like RoboCat, which generalizes across robotic embodiments for goal-conditioned tasks [2], and RT-X , which handles language-conditioned manipulation across robots [1].

Octo, an open-source generalist robot policy, represents a significant advancement in this domain [6]. Built on a transformer-based architecture, Octo integrates diverse inputs, including language instructions, goal images, and proprioceptive data, into a unified tokenized format. Trained on the data from the Open X-Embodiment dataset, the largest robotics dataset to date, Octo sets a new standard for scalability and adaptability.

One of Octo's key innovations is its diffusion-based action generation mechanism, which generates smooth, continuous control commands. This approach ensures robust multi-modal action handling, allowing the model to excel in both zero-shot generalization and efficient fine-tuning. Experiments demonstrate Octo's ability to adapt to unseen robots and tasks using as few as 100 demonstrations, outperforming state-of-the-art models like RT-1-X and VC-1.

This review critically evaluates Octo's contributions to the field of GRPs, comparing it with related approaches and exploring its applications, strengths, and limitations. By providing a comprehensive analysis, we aim to highlight how Octo advances the vision of scalable, adaptable, and open-source robotics.

### A. Key Contributions

In general, the key contributions of the Octo paper are as follows:

- Introduction of an open-source transformer-based Generalist Robot Policy.
- Utilization of data from the Open X-Embodiment dataset, the largest robotics dataset to date.
- Implementation of a unified tokenized representation for multi-modal inputs.
- Adoption of a diffusion-based action head for smooth and continuous robot control.
- Demonstration of strong zero-shot generalization and efficient fine-tuning.

## II. Background and Related work

The field of Generalist Robot Policies (GRPs) has evolved to address the need for adaptable, scalable, and efficient robotic systems. Unlike traditional task-specific approaches, GRPs aim to unify control for diverse tasks, sensor configurations, and robot embodiments. This section provides an overview of key developments in GRPs and positions Octo within this context.

### A. Generalist Robot Policies

GRPs are inspired by foundation models in natural language processing and computer vision, such as GPT and ViT [5], which leverage large-scale pretraining to generalize across tasks. Similarly, GRPs aim to harness diverse robotic datasets to train policies that can be fine-tuned for new applications with minimal data.

Several notable GRPs have emerged in recent years:

1) RoboCat: Focuses on goal-conditioned tasks across multiple robotic embodiments. While effective, it requires significant retraining when adapting to new domains [2].
2) RT-X: Specializes in language-conditioned manipulation but is limited by its reliance on fixed input modalities and lack of open-source availability [1].
3) GNM: Addresses generalization in navigation tasks but does not extend to complex manipulation [3].

These models often suffer from key limitations:

1) Restricted input flexibility, such as reliance on a single sensor modality [8].
2) Limited scalability to novel robots and tasks without extensive retraining [1].
3) Lack of open-source availability, hindering reproducibility and community-driven improvements.

Octo represents a paradigm shift in the GRP landscape, addressing these limitations through its innovative design and comprehensive dataset. Octo builds on the strengths of its predecessors while addressing their shortcomings. Table 1 below summarizes its advantages over other GRPs:

Table 1: Comparison of Octo with Prior GRPs

| **Model** | **Input Flexibility** | **Action Space** | **Open Source** |
|---|---|---|---|
| RoboCat | Moderate | Goal-conditioned | No |
| RT-X | Low | Language-conditioned | No |
| GNM | Moderate | Navigation-only | Partial |
| Octo | High | Multi-modal (diffusion) | Yes |

Before discussing Octo's architecture, it is important to introduce key concepts that form the foundation of this work. These include diffusion processes, action chunking, zero-shot and few-shot learning, causal masked blockwise attention, and the dataset used for training. A clear grasp of these topics will help readers better appreciate the design choices and contributions of Octo.

## *B. Zero-Shot and Few-Shot Learning*

Zero-shot learning refers to the model's ability to perform a new task it has never seen during training, while few-shot learning involves adapting to new tasks with a small number of demonstrations. Octo excels in both paradigms, requiring as few as 100 demonstrations to fine-tune for novel tasks and achieving high performance even in zero-shot scenarios [6], [10].

## *C. Causal Masked Blockwise Attention*

Causal masked blockwise attention is a mechanism that allows transformer models to process long sequences efficiently while maintaining the structure and order of the data [11]. To fully understand this, let's break down each component—causal, masked, and blockwise—before explaining how they work together in Octo's architecture.

The term causal refers to the sequential nature of data processing, where each step depends only on information from the past and not the future. For example, in a robot arm moving to a target, causal processing ensures that the model uses the current and past states to predict the next action without accessing information about future states, which would be unrealistic in real-time scenarios.

The term masked refers to the selective ignoring or ”masking out” of irrelevant or unavailable information during the attention process. In the context of causal attention, masking ensures that the model does not attend to tokens representing future time steps.

The blockwise aspect refers to dividing the input sequence into smaller, manageable chunks or blocks instead of processing the entire sequence at once.

## *D. Action Chunking*

Action chunking is a method where a sequence of future actions, called a ”chunk,” is predicted at once instead of generating single-step actions one at a time [6], [12]. In action chunking, the system predicts a chunk of actions that correspond to a specific time horizon. Each chunk contains multiple control commands, such as the robot's joint positions, velocities, or end-effector movements.

## *E. Diffusion Model*

Diffusion models are models where information is gradually corrupted with noise and then reconstructed by reversing this process. In robotics, diffusion models are used for action generation by adding noise to initial predictions and iteratively refining them to produce smooth and robust control signals [4], [10].

Diffusion models are central to how Octo generates actions, allowing it to produce smooth, precise, and adaptable movement sequences [4], [10]. These models work by using a two phase process: a ”forward” process that adds noise to the data and a ”backward” process that removes it.

1) Forward Process (Adding Noise): Imagine you have a clear picture. In the forward process, you're essentially making the picture progressively blurrier by gradually adding noise, step-by-step, until it becomes almost unrecognizable [4], like static on a TV screen. In Octo's case, the ”picture” is a sequence of actions the robot needs to take.
2) Backward Process (Removing Noise): The backward process is the reverse. Starting from the noisy, almost random data, the model gradually removes the noise, step-by-step [4]. It's like cleaning up the blurry picture, refining it until the original clear image is restored. For Octo, this means refining the noisy data back into a coherent and precise sequence of actions.

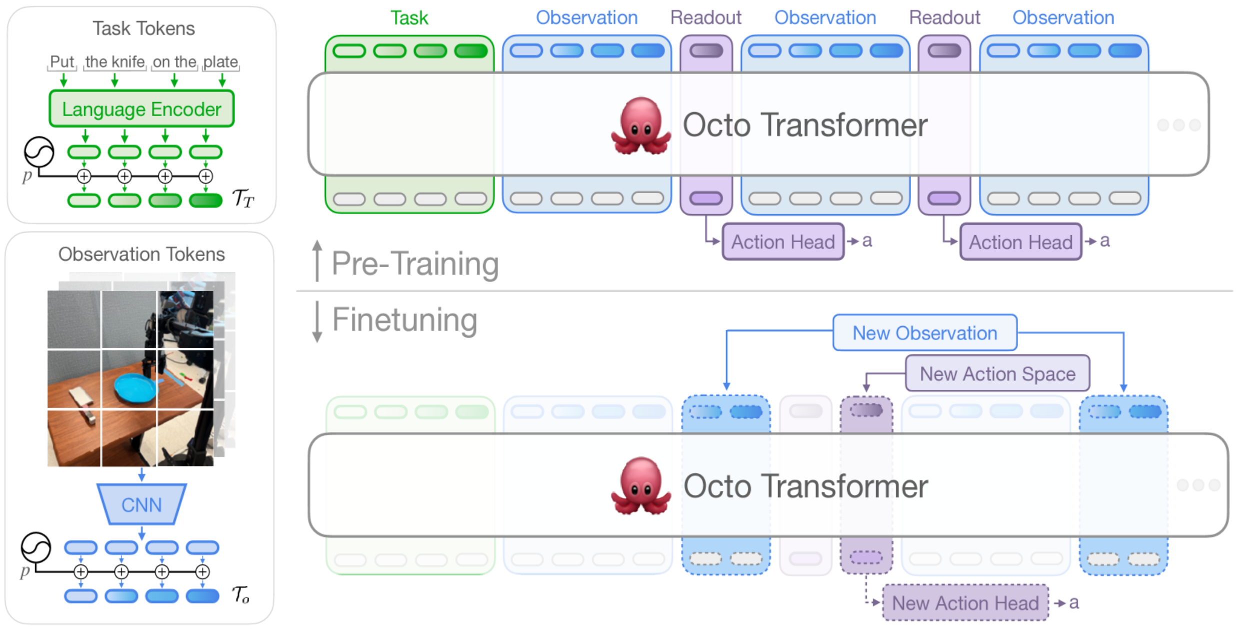


Figure 1: **Octo's model architecture.** Task descriptions (green) and input observations (blue) are turned into tokens using a language model and a lightweight CNN. These tokens are processed by the transformer, which creates readout tokens (purple) that produce actions through output heads. The blockwise attention allows adding new inputs (blue, dashed) or outputs (purple, dashed), like observations or action spaces, during fine-tuning without changing the pretrained model.

## III. Model Architecture

The architecture of Octo, Figure 1, is designed to integrate diverse input modalities, efficiently process large sequences of data, and generate smooth, continuous control commands [5]. It leverages a transformer-based backbone with several novel components to achieve these goals. This section provides a detailed overview of Octo's architecture, focusing on its unified token representation, blockwise masked attention, readout tokens, and diffusion-based action head.

### A. Unified Token Representation

Octo begins by converting diverse input data into a unified tokenized format that can be processed by the transformer model [5]. Figure 2 illustrates this tokenization process. This unification enables Octo to handle multiple input types (language, vision, and proprioceptive data) within a single framework.

### B. Input Types

*1) Language Input:* Task instructions in natural language are tokenized using the T5-Base model [7]. These tokens are embedded into a continuous representation, preserving semantic meaning for downstream processing.

*2) Visual Input:* Images from wrist-mounted and third-person cameras are divided into patches (e.g., $16 \times 16$ pixels) and encoded into tokens using a lightweight convolutional network (CNN). These visual tokens capture spatial information, allowing the model to understand the robot's environment [5].

*3) Proprioceptive Input:* Data from the robot's sensors, such as joint positions and velocities, are directly mapped into token embeddings. These tokens provide the model with the robot's internal state, essential for precise control [6].

### C. Blockwise Masked Attention

Octo's transformer backbone employs blockwise masked attention, which enables efficient processing of long token sequences without sacrificing contextual understanding.

1) Blockwise Processing: The input token sequence is divided into smaller, manageable blocks. Attention is applied within each block, focusing on local context while reducing computational overhead [11].

2) Causal Masked Attention: Ensures that each token only attends to previous tokens in the sequence, maintaining the temporal order of data [11]. This approach prevents the model from accessing ”future” information, which is critical for real-time robotic control.

Blockwise masked attention allows Octo to scale to longer sequences of inputs, such as high-resolution images or extended proprioceptive data, while maintaining computational efficiency.

### D. Readout Tokens

Readout tokens are a unique feature of Octo's architecture, designed to summarize the task and observational context for action generation. Figure 2 provides a visual breakdown of the token process flow. These are learnable tokens added to the input sequence during training. Unlike regular tokens, readout

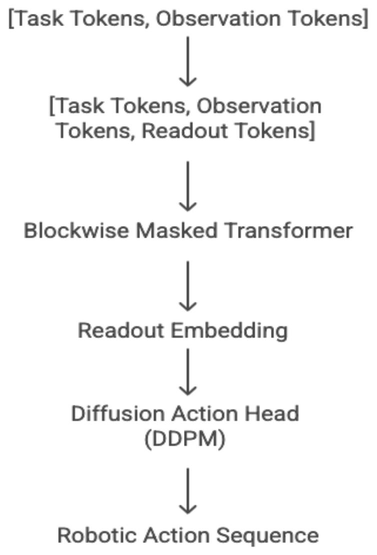

[Task Tokens, Observation Tokens]
[Task Tokens, Observation Tokens, Readout Tokens]
Blockwise Masked Transformer
Readout Embedding
Diffusion Action Head (DDPM)
Robotic Action Sequence

Figure 2: **Token Process Flow.** Task, observation, and readout tokens processed through Octo transformer.

tokens do not influence other tokens in the sequence. Instead, they act as "passive listeners," attending to the entire input sequence to extract high-level summaries. After processing through the transformer backbone, the embeddings of these tokens serve as compact representations of the task and environment, guiding the action generation process [6].

## *E. Diffusion-Based Action Head*

The diffusion-based action head is the final stage of Octo's architecture, responsible for generating smooth and continuous control commands [4], [10]. The readout tokens are passed to the diffusion model, which predicts a sequence of future actions (action chunks) over a fixed time horizon. Each action chunk includes control parameters such as the end-effector position, orientation, and gripper state. These sequences of actions are analogous to the time horizons used in model predictive control (MPC), where actions are optimized over a finite prediction window to ensure smooth and robust control. The diffusion model iteratively refines noisy initial predictions to produce high-quality control signals. This two-phase process (forward noise addition and backward denoising) ensures that the generated actions are smooth and robust. The key feature is the denoising process, described mathematically below:

$$x^{k-1} = \alpha \left( x^k - \gamma \epsilon_\theta(x^k, e, k) \right) + \mathcal{N}(0, \sigma^2 I) \tag{1}$$

Where:

- $x^k$: The noisy action state at diffusion step $k$.
- $\epsilon_\theta$: The learned denoising network parameterized by $\theta$.
- $e$: Task and environment-specific embeddings.
- $\alpha, \gamma$: Hyperparameters guiding the noise scaling.
- $\mathcal{N}(0, \sigma^2 I)$: Small Gaussian noise added for stability and smoothness.

This equation illustrates how the diffusion process iteratively transforms noisy intermediate states ($x^k$) into a denoised, high-quality action ($x^{k-1}$). The Gaussian noise term $\mathcal{N}(0, \sigma^2 I)$ ensures robustness and prevents overfitting to specific data patterns. Unlike traditional models that predict discrete actions, Octo generates continuous-valued control commands, enabling natural and precise robot movements.

## *F. Data Selection*

The success of Octo is rooted in its training on the Open X-Embodiment dataset. The Open X-Embodiment dataset, the largest open-source dataset for robotic manipulation, contains over 1 million demonstrations spanning 22 different robot embodiments, including robotic arms and bi-manual robots. This dataset covers a diverse range of tasks, environments, and sensory modalities, providing a strong foundation for training generalist robot policies. From this extensive dataset, the Octo Data Mix was curated by selecting 800,000 high-quality trajectories across 25 datasets specifically designed for training generalist robot policies. These selected trajectories were used to train the Octo model. The variety of data ensures that Octo can generalize across unseen tasks and adapt to new environments with minimal additional training. Datasets with low-resolution images, overly repetitive tasks, or insufficient sensory data were excluded to maintain relevance [6].

## *G. Pretraining*

By utilizing multi-modal inputs, including camera views, language annotations, and proprioceptive data from Octo Data Mix, Octo learns shared representations for these diverse modalities [6]. The Pretraining process fosters generalization, reduces overfitting to specific robot types or tasks, and allows for efficient fine tuning with minimal additional data.

## *H. Fine Tuning*

Fine Tuning builds upon the foundation of Octo's pre-trained knowledge, avoiding the need to retrain the entire model from scratch for each new scenario. A key advantage of Octo's fine tuning approach is its data efficiency. Often, as few as 100 demonstrations are sufficient to adapt the model to a new task [6], [9]. This is possible because the model starts with weights initialized from pre-training, allowing it to leverage previously learned knowledge and generalize effectively. Empirical results presented in the paper confirm that this approach leads to faster convergence and higher success rates when compared to training a model from scratch.

The architecture of Octo is specifically designed to support modularity through the use of lightweight adapter modules. This modularity is key to the fine-tuning process.

## IV. Experiments and Results

The experimental evaluation of Octo demonstrates its ability to generalize across tasks, adapt to new environments, and achieve competitive performance with state-of-the-art models.

This section presents the key findings from the experiments conducted in the Octo paper, focusing on zero-shot generalization, fine-tuning efficiency, and the impact of architectural choices.

The experiments aim to address the following key questions:

1) Can Octo handle diverse robots and tasks without retraining?
2) Does Octo's pretraining enable efficient fine-tuning compared to training from scratch?
3) Which design choices in Octo contribute most to its effectiveness?

These questions guide the evaluation of Octo's generalization, adaptability, and architectural strengths. The following subsections provide detailed answers to each.

## *A. Zero-Shot Generalization*

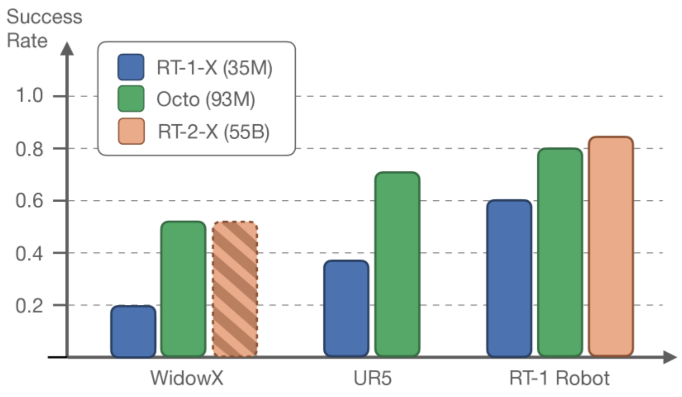


Figure 3: **Zero-Shot Capabilities** [6].

Octo's zero-shot generalization capabilities were rigorously evaluated to assess its ability to perform tasks on unseen robots and environments. These experiments are crucial for demonstrating the model's robustness and adaptability to real-world scenarios. Octo was tested on manipulation, navigation, and goal-conditioned tasks using robotic embodiments not present in its pretraining data. Examples include pick-and-place tasks with a new robot arm and navigating unfamiliar environments [6]. Figure 3 presents a graphical summary of these results. The results show that Octo outperforms RT-1-X across all tested robots, demonstrating its ability to generalize effectively to unseen environments. Octo performs comparably to RT-2-X (55B) on the WidowX and RT-1 Robot tasks. Additionally, the variation in performance across different robots indicates that embodiment differences influence generalization capabilities, emphasizing the importance of diverse pretraining data [4], [10].

Figure 4 demonstrates that increasing model size leads to improved zero-shot success rates, highlighting the effectiveness of scaling in generalist robot policies. Both UR5 and WidowX tasks show a steady upward trend, indicating that Octo benefits from larger model capacities. This consistent performance gain suggests that scaling not only enhances task

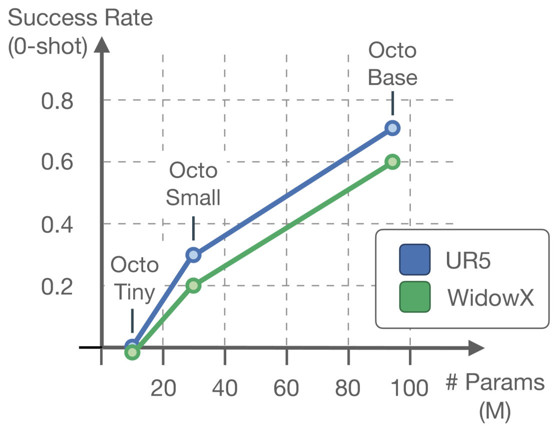


Figure 4: **Model Scaling Results.** The performance of Octo improves with larger model sizes. Success rates are averaged over 10 trials on one language-conditioned task per robot. Octo-Tiny (10M), Octo-Small (27M), and Octo-Base (93M). [6]

execution but also strengthens Octo's ability to generalize across different robotic platforms and environments.

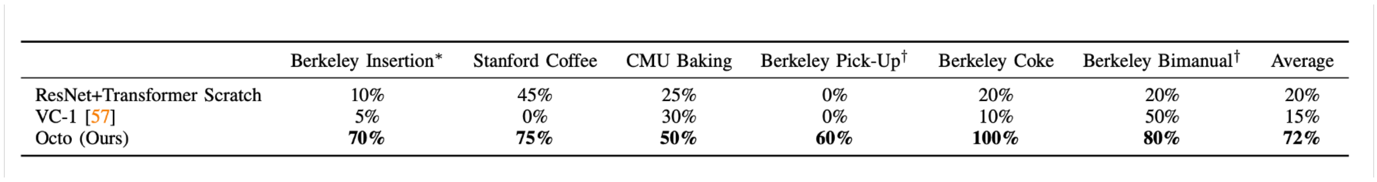
## *B. Few-Shot Fine-Tuning*

Octo's ability to adapt to new tasks and robots with minimal data was evaluated through few-shot fine-tuning experiments, a crucial capability for real-world deployment where collecting large datasets is often impractical. Octo was fine-tuned with approximately 100 additional demonstrations per task or robot to assess its robustness. The results, summarized in Table 2, show that fine-tuning Octo yields superior policies compared to training from scratch or using pretrained VC-1 weights. Across six evaluation setups, Octo outperforms the baseline in all the setups [6]. The effectiveness of Octo's few-shot fine-tuning could be directly attributed to its diverse pretraining dataset. These results demonstrate Octo's ability to rapidly adapt to novel tasks and robots with minimal data.

## *C. Architectural Ablations*

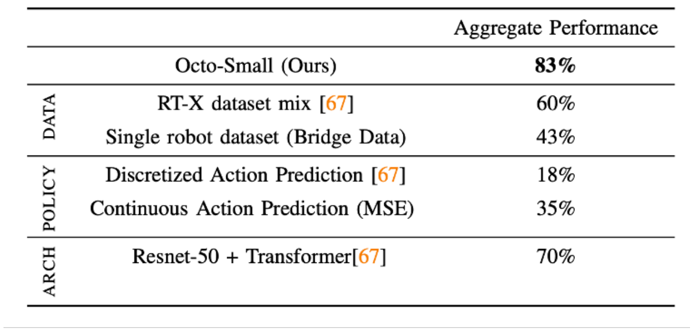
To understand the contribution of individual design choices to Octo's performance, the authors conducted several ablation studies such as replacing the diffusion-based action head with a simpler MSE-based head.

The results in Table 3 indicate that ViT, diffusion-based action heads, and a wide dataset mixture significantly contribute to Octo's success. Simpler architectures, such as MSE-based action prediction or ResNet-based models, result in lower performance, highlighting the importance of advanced model design. Additionally, training on a diverse dataset with multiple embodiments leads to better generalization compared to single-robot datasets. These findings reinforce the need for both well-structured architectures and broad, multi-modal training data to achieve optimal performance in generalist robot policies [6].



| | Berkeley Insertion* | Stanford Coffee | CMU Baking | Berkeley Pick-Up† | Berkeley Coke | Berkeley Bimanual† | Average |
|---|---|---|---|---|---|---|---|
| ResNet+Transformer Scratch | 10% | 45% | 25% | 0% | 20% | 20% | 20% |
| VC-1 [57] | 5% | 0% | 30% | 0% | 10% | 50% | 15% |
| Octo (Ours) | **70%** | **75%** | **50%** | **60%** | **100%** | **80%** | **72%** |

Table 2: **Finetuning results.** Octo compared with 2 baseline models. ResNet+Transformer Scratch serves as a ”basic” or ”naive” baseline. VC-1 weights are combined with a separate action decoder (e.g., an MLP) [6]



| | | Aggregate Performance |
|---|---|---|
| | Octo-Small (Ours) | **83%** |
| DATA | RT-X dataset mix [67] | 60% |
| | Single robot dataset (Bridge Data) | 43% |
| POLICY | Discretized Action Prediction [67] | 18% |
| | Continuous Action Prediction (MSE) | 35% |
| ARCH | Resnet-50 + Transformer[67] | 70% |

Table 3: **Model Ablations.** Best performance when using the ViT architecture, diffusion action head, and wide training data mixture. Success rates are averaged over 40 trials across 2 language-conditioned tasks and 2 goal-conditioned tasks [6]

## V. Discussion

Experimental results highlight Octo’s strengths in generalization, data-efficient adaptation, and fine-tuning. Firstly, Octo demonstrates robust generalization, excelling at performing tasks across a variety of robots and environments without additional training. This zero-shot capability, stemming from its training on the diverse Open X-Embodiment dataset and its unified token representation, is crucial for real-world applicability where encountering novel situations is inevitable. Secondly, Octo exhibits data-efficient adaptation through fine tuning. The ability to adapt to new tasks and robots with as few as 100 demonstrations significantly reduces the cost and time required for deployment, a major advantage in practical applications where data collection is often a bottleneck. Finally, Octo’s innovative architecture, particularly the combination of diffusion-based action heads and blockwise masked attention [4], [10], ensures smooth action generation and efficient processing of large, multi-modal input sequences. These design choices are key to Octo’s ability to handle complex tasks with precision.

While Octo performs well, certain limitations require further study. Performance in language-conditioned tasks lags behind that of image-conditioned tasks, indicating a need for improved alignment between language and visual inputs during training. This could involve increasing the proportion of language-annotated data in the pretraining dataset or exploring more sophisticated multi-modal fusion techniques. Furthermore, while Octo handles multi-sensor fusion reasonably well, tasks requiring the integration of underrepresented sensor modalities, such as combining wrist camera and force-torque data, revealed performance gaps. This suggests a need for more balanced training datasets that adequately represent the diversity of real-world sensor inputs. Finally, while Octo performs well in general tasks, domain-specific adaptation to highly specialized environments or tasks may still require significant domain-specific data and careful hyperparameter adjustments, highlighting an area for future improvement.

The findings presented in the Octo paper open up several promising avenues for future research [6]. Improving language-conditioned policies, as mentioned earlier, is a key area. Another is expanding sensor modalities by incorporating inputs like tactile feedback and depth sensing, which could enhance Octo’s adaptability to real-world scenarios demanding fine-grained control. Handling real-world variability is also crucial; training on datasets with sub-optimal or noisy demonstrations could enable Octo to better cope with the unpredictability of real-world environments. Finally, scaling model size and investigating the impact of larger model variants on zero-shot and fine-tuning performance could further enhance Octo’s capabilities and scalability.

In conclusion, Octo represents a significant leap forward in the development of generalist robot policies. Its strengths in generalization, adaptability, and efficiency, coupled with its innovative architecture, make it a promising foundation for future research and real-world applications. While challenges remain in areas like language conditioning and handling underrepresented sensor modalities, addressing these limitations will pave the way for even more robust and versatile robotic agents capable of operating seamlessly in complex, dynamic environments. The open-sourcing of Octo will undoubtedly accelerate progress in this exciting field, fostering a collaborative effort towards building truly generalist robots.

## References

[1] Open X-Embodiment Collaboration, “Open X-Embodiment: Robotic learning datasets and RT-X models,” *arXiv preprint*, arXiv:2310.08864, 2023. [Online]. Available: [https://arxiv.org/abs/2310.08864](https://arxiv.org/abs/2310.08864).

[2] K. Bousmalis *et al.*, “RoboCat: A self-improving foundation agent for robotic manipulation,” *arXiv preprint*, arXiv:2306.11706, 2023.

[3] D. Shah *et al.*, “GNM: A general navigation model to drive any robot,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7226–7233.

[4] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 6840–6851, 2020.

[5] A. Dosovitskiy *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint*, arXiv:2010.11929, 2020.

[6] D. Ghosh *et al.*, “Octo: An open-source generalist robot policy,” *arXiv preprint*, arXiv:2405.12213, 2024. [Online]. Available: [https://octo-models.github.io/](https://octo-models.github.io/).

[7] C. Raffel *et al.*, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[8] S. Dasari *et al.*, “Robonet: Large-scale multi-robot learning,” in *Conf. Robot Learn.*, PMLR, 2020, pp. 885–897.

[9] A. Brohan *et al.*, “RT-1: Robotics transformer for real-world control at scale,” *arXiv preprint*, arXiv:2212.06817, 2022.

[10] C. Chi *et al.*, “Diffusion policy: Visuomotor policy learning via action diffusion,” in *Proc. Robot. Sci. Syst.*, 2023.

[11] W. Chen *et al.*, “Vision-language models provide promptable representations for reinforcement learning,” *arXiv preprint*, arXiv:2402.02651, 2024

[12] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, ”Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware,” arXiv preprint arXiv:2304.13705, Apr. 2023. [Online]. Available: https://arxiv.org/abs/2304.13705