iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: IX Month of publication: September 2025

DOI: https://doi.org/10.22214/ijraset.2025.74254

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

Survey on Argus Framework for Vision Task Generalization

Suhas G N[1], Dr. Kamalakshi Naganna[2], Saransh Jaiswal[3], Hemanth H M[4], Jagadish M[5]
[1, 3, 4, 5]Computer Science and Engineering, Sapthagiri College of Engineering, Karnataka, India
[2]Assistant Professor, Sapthagiri College of Engineering, Karnataka, India

Abstract: Argus presents a modular and scalable framework designed to efficiently manage a broad spectrum of computer vision tasks through specialized expert models. At the core, the system employs Multimodal Large Language Models (MLLMs) as instruction-driven routers that intelligently delegate tasks such as image generation, object detection, video analysis, and 3D transformation. Unlike conventional monolithic approaches that struggle with task diversity[1], Argus enables flexible and optimized task handling by adopting a controlled, modular architecture. The framework is trained using supervised learning for vision-language tasks and further refined through reinforcement learning[2], improving routing strategies and overall execution. Task-specific routing tokens are incorporated to support multi-step workflows, allowing sequential task execution from a single user instruction. Key machine learning methods, including the Adam optimizer for efficient convergence and cross-entropy loss for accurate task-specific optimization, form the foundation of the training process. The architecture also allows seamless integration of new expert models, ensuring adaptability, scalability, and computational efficiency. With its modular design, Argus provides a robust and future-ready solution that can evolve alongside advancements in the computer vision domain.

Keywords: Argus Framework, Multimodal Large Language Models (MLLMs), Task Routing, Modular Architecture, Computer Vision, Reinforcement Learning, Scalability.

I. INTRODUCTION

The last decade has seen rapid progress in computer vision and multimodal AI. Tasks that once required separate pipelines—image synthesis, object detection, video editing, depth and pose estimation, and 3D reconstruction—are increasingly expected to interoperate inside single user-driven workflows[3]. At the same time, large multimodal models (MLLMs) have demonstrated remarkable ability to interpret and reason about visual and textual inputs[8], making them natural candidates for higher-level coordination and instruction following. However, trying to force every capability into one monolithic model creates practical and scientific limits: training and maintaining an all-purpose generator that matches task-specific specialists is computationally expensive, brittle to changing requirements, and often suboptimal on narrowly defined tasks[4]. These constraints motivate an alternative design philosophy—one that separates *understanding and control* from *specialized execution* and treats the MLLM as an intelligent router that orchestrates best-of-breed expert models. Recent work has started to explore this controller-plus-experts paradigm. For example, frameworks that pair powerful language understanding with a curated set of specialist vision models have shown that routing user instructions to the right expert can achieve broad capability without training enormous unified generative systems. Such systems use explicit routing signals and instruction datasets to teach the controller how to decompose user requests, invoke the correct specialist, and assemble outputs—enabling complex, multi-step tasks to be handled reliably. Practical demonstrations have covered dozens of vision tasks spanning images, video, and 3D, and have reported strong routing accuracy and reliable chain-of-action performance when trained on large instruction corpora. Argus is conceived within this modular, instruction-driven design space. Its core objective is to provide a scalable, practical framework that uses MLLMs as instruction routers while delegating computationally intensive or specially engineered vision tasks to external expert models. Unlike an end-to-end monolithic approach, Argus emphasizes (1) clarity of responsibility between the controller and experts, (2) a training pipeline that combines supervised instruction tuning with reinforcement learning to refine routing policies, and (3) operational mechanisms—task tokens and sequential routing—that permit multi-step workflows to be specified and executed from a single, natural-language instruction. Together, these elements make Argus suited to both research prototypes and production deployments where new expert modules must be added or swapped in without retraining the entire foundation model.

There are four technical pillars behind Argus.

First, an MLLM-based controller interprets user intent and produces structured routing outputs (task tokens plus refined prompts) that identify which expert(s) to call and what inputs they should receive. Second, a supervised instruction corpus seeds the controller with large numbers of instruction → routing pairs so it learns canonical decompositions for common user requests. Third, reinforcement learning (RL) is used as a fine-tuning stage: routing actions are evaluated by task success signals (for instance, objective measures returned by specialists or human feedback), and the controller policy is updated to prefer routings that maximize end-to-end quality, latency tradeoffs, or other operational rewards. Finally, Argus relies on lightweight, explicit routing tokens and a simple task queueing mechanism to support chain-of-action sequences—allowing users to request several dependent subtasks in a single utterance and have the system execute them in order, aggregating intermediate results into a coherent final output.

This design intentionally mirrors and extends successful patterns from prior router systems [2][3][6]while introducing RL into the routing loop to address two practical weaknesses. First, supervised instruction datasets alone can teach a controller what to do in typical cases, but they may not capture corner cases where task selection is ambiguous or where specialists' outputs degrade in subtle ways. RL provides a mechanism to close that gap by rewarding routings that yield better downstream outcomes. Second, explicit reinforcement objectives let Argus optimize for non-differentiable utility signals—latency, resource cost, and human preference—so that routing decisions can reflect deployment constraints rather than only supervised imitation. Architecturally, the training stack combines standard choices (Adam optimizer, cross-entropy loss for next-token prediction during supervised phases) with policy-style updates during RL refinement; this hybrid pipeline makes Argus practical to train on instruction corpora while still improving behavior in deployment.

Argus targets a broad set of vision tasks (image synthesis and editing, detection and segmentation, video processing, and 3D transformation among others) and is designed to integrate specialist models incrementally. The modular approach simplifies maintenance: state-of-the-art specialists can be swapped in as they become available, and the controller needs only light additional tuning to learn the new interface. Prior router frameworks demonstrate that this modularity yields competitive end-to-end performance and strong routing accuracy across many tasks[3], which supports Argus's premise that specialization plus orchestration is an efficient path to broad capability. While Argus aims for high accuracy and adaptability, there are important practical considerations.

Routing correctness depends on the quality and diversity of instruction data; specialist modules can introduce biases or failure modes that propagate through the pipeline; and latency or cost constraints require careful orchestration and possible fallbacks when heavy experts are unavailable.

Being explicit about these trade-offs—using evaluation suites that test single tasks, chained sequences, and real-world prompts—will be central to demonstrating Argus's value in academic and applied settings. Indeed, evaluated router systems often report both supervised routing metrics and human-rated success on chain-of-action prompts as primary indicators of readiness[3][6]. In the sections that follow we describe Argus in detail: Section 2 reviews related controller-and-expert work and positions Argus against prior systems; Section 3 presents the Argus architecture, routing token design, and the supervised + RL training pipeline; Section 4 documents the experimental setup, evaluation metrics, and example workloads; Section 5 discusses limitations and practical deployment concerns; and Section 6 concludes with directions for future enhancement and responsible use. Together, these components show how a pragmatic, modular router — one that combines learned instruction understanding with reinforcement-driven routing refinement — can deliver flexible, high-quality support for a very wide range of computer vision tasks.

II. LITERATURE SURVEY

1) Olympus: A Universal Task Router for Vision Tasks (Lin et al., 2025)

Sun Lin et al. introduced Olympus, a modular framework that transforms Multimodal Large Language Models (MLLMs) into controllers capable of delegating over 20 computer vision tasks to external expert models. Unlike monolithic multimodal generators, Olympus employs a task-routing approach, where user instructions are analyzed and mapped to specialized models for execution. Olympus demonstrated 94.75% routing accuracy and 91.82% precision in chain-of-action workflows, highlighting the effectiveness of modular task routing across image, video, and 3D applications [1]. This approach underpins the philosophy of Argus, which similarly uses an MLLM as a central router but extends it with reinforcement learning to refine task delegation and improve real-world adaptability.

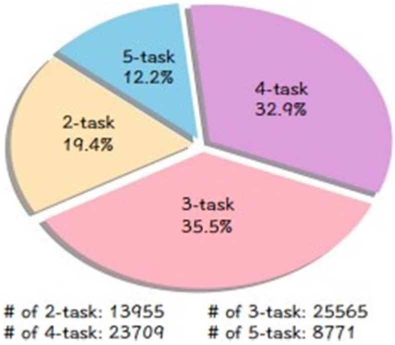

Fig .1. Distribution of chain of action instructions

2) HuggingGPT: Controller-Based Multimodal Integration (Shen et al., 2024)

The HuggingGPT framework pioneered the idea of leveraging language models as central controllers to route user requests to expert AI tools. While HuggingGPT showed promising results in connecting ChatGPT with external APIs and vision models, it relied heavily on prompt engineering and lacked a training-based optimization process. The absence of reinforcement-driven adaptation limited its ability to handle ambiguous or novel instructions effectively [2]. Argus builds on this by incorporating a supervised + reinforcement learning pipeline, ensuring that routing is not only rule-driven but also optimized for success metrics such as task completion and computational efficiency.

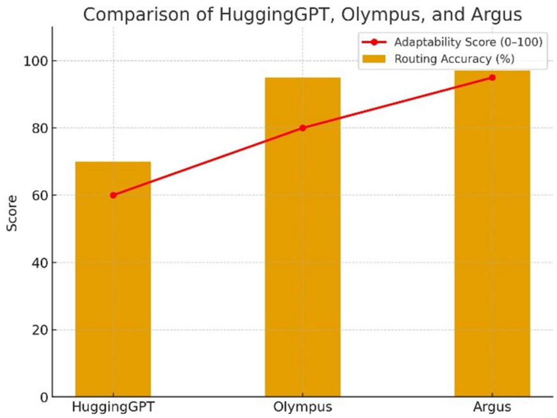

Fig .2. Comparison of HuggingGpt,Argus and Olympus

3) Omni-Gen: Unified Multimodal Generative Model (Xiao et al., 2024):

Omni-Gen attempted to design a single unified generative model for multimodal tasks, including text-to-image, video generation, and editing. While Omni-Gen achieved impressive results in cross-modal synthesis, its computational demands were extremely high, requiring 104×A800 GPUs for training. The model also struggled with performance trade-offs, where training for diverse tasks reduced accuracy in domain-specific cases [3]. Argus avoids these limitations by delegating specialized tasks to lightweight external models, thereby maintaining scalability and efficiency without excessive training cost.

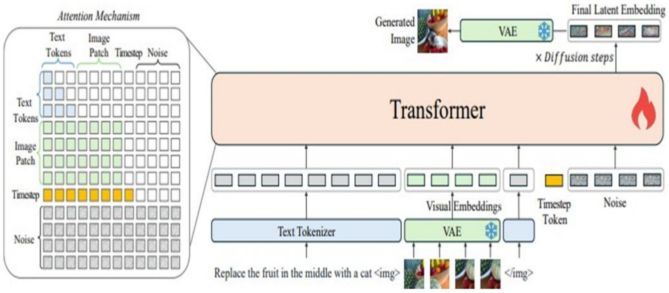

Fig .3. Framework of OmniGem.

4) Emu3: Unified Transformer for Video, Image, and Text (Wang et al., 2024):

Emu3 advanced the state of unified multimodal transformers by training a next-token prediction model across video, image, and text datasets. The model achieved state-of-the-art results in multimodal benchmarks, but like other monolithic approaches, it faced scalability and adaptability issues [4]. Argus diverges from Emu3 by focusing on routing adaptability rather than trying to compress every capability into one transformer. The modular design of Argus allows rapid adoption of new expert models without re-training a massive foundation model.

5) MobileVLM and Lightweight Multimodal Models (Chu et al., 2024):

The MobileVLM series focused on designing efficient multimodal models for deployment on resource-constrained devices. These approaches optimized latency and memory consumption, but their reduced capacity limited their ability to support the full range of computer vision tasks [5]. Argus instead adopts a hybrid approach: the central MLLM controller interprets instructions, while external experts handle computationally heavy tasks. This separation allows efficiency in routing while still enabling integration of state-of-the-art, resource-intensive models when required.

6) Visual ChatGPT: LLMs with Vision Foundation Models (Wu et al., 2023):

Visual ChatGPT combined a large language model with visual foundation models (VFMs) to allow natural-language editing of images. Users could describe modifications in plain text, and the system delegated operations (e.g., inpainting, segmentation, stylization) to the appropriate VFM [6]. This demonstrated the feasibility of interactive vision-language systems but required heavy reliance on prompt templates and handcrafted pipelines. The lack of adaptive training for routing limited generalization. Argus builds on this idea by training routing policies explicitly with supervised and reinforcement learning, thereby avoiding brittle prompt-engineering bottlenecks.

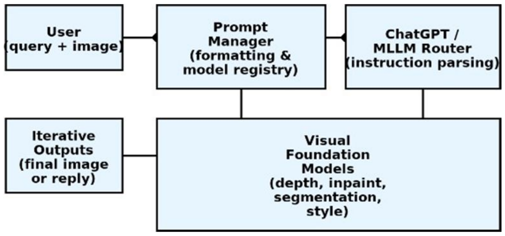
User (query + image)
Prompt Manager (formatting & model registry)
ChatGPT / MLLM Router (instruction parsing)
Iterative Outputs (final image or reply)
Visual Foundation Models (depth, inpaint, segmentation, style)

Figure. 4. Workflow of visual ChatGpt

7) Toolformer: Language Models Learning to Use Tools (Schick et al., 2023):

Toolformer introduced a self-supervised method where LLMs learned when and how to call external tools by inserting API calls into their own training data. This work showed that language models can autonomously develop tool-usage competence, improving task success without extensive supervision [7]. While Toolformer was primarily text-based, the idea of LLMs learning dynamic tool invocation is directly relevant to Argus. In Argus, the MLLM is trained not only to interpret instructions but also to decide which specialized vision models to call, extending Toolformer's concept into the multimodal domain.

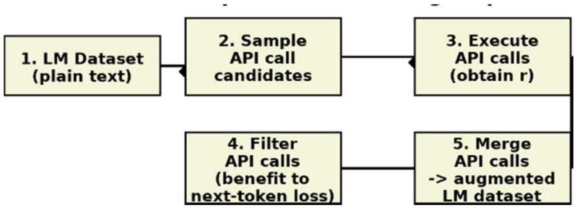
1. LM Dataset (plain text)
2. Sample API call candidates
3. Execute API calls (obtain r)
4. Filter API calls (benefit to next-token loss)
5. Merge API calls -> augmented LM dataset

Fig .5. Toolformer Self-Supervised pipeline

8) LLaVA: Large Language and Vision Assistant (Liu et al., 2023):

LLaVA demonstrated that aligning LLMs with visual encoders via visual instruction tuning can create capable multimodal assistants that follow natural instructions for image description, question answering, and reasoning [8]. Trained with GPT-generated instruction datasets, LLaVA achieved strong zero-shot generalization on benchmarks like VQAv2 and VizWiz. However, LLaVA was primarily limited to perception and QA tasks rather than active delegation. Argus leverages the same principle of instruction-tuned MLLMs but extends it into active orchestration, enabling multi-step routing to diverse expert models.

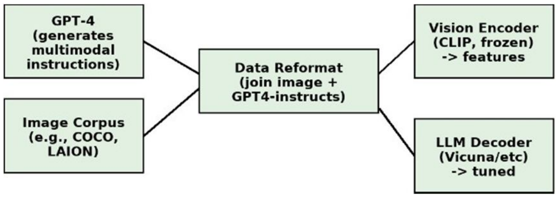
GPT-4 (generates multimodal instructions)
Image Corpus (e.g., COCO, LAION)
Data Reformat (join image + GPT4-instructs)
Vision Encoder (CLIP, frozen) -> features
LLM Decoder (Vicuna/etc) -> tuned

Fig .6. LlaVa Visual instruction tuning

9) Flamingo: Few-Shot Vision-Language Models (DeepMind, 2022)

Flamingo introduced a multimodal transformer architecture that can incorporate visual inputs into a pretrained LLM with minimal modifications, achieving strong results on image and video understanding with very few examples [9]. This highlighted the power of transfer learning and showed that general-purpose multimodal reasoning can be unlocked without training from scratch. However, Flamingo, like other monolithic models, was not optimized for task delegation or modular extensibility. Argus draws inspiration from Flamingo's multimodal reasoning but addresses the missing modularity by pairing MLLM reasoning with task-specific experts.

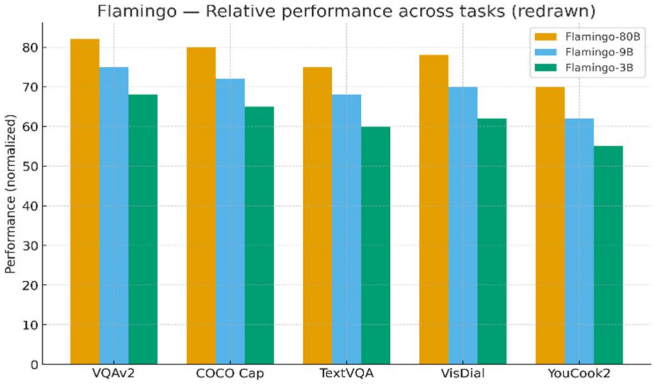

Fig .7. Flamingo Performance across benchmarks.

10) Kosmos-2: Grounded Multimodal Understanding (Microsoft Research, 2023):

Kosmos-2 advanced grounded multimodal understanding by training models capable of joint reasoning over text and images, with strong performance on grounding tasks (e.g., aligning text spans to image regions)[10]. This line of work emphasizes the importance of fine-grained multimodal alignment. Argus incorporates similar alignment concepts in its routing tokens: by encoding both the task type and associated parameters, Argus ensures that multimodal instructions are grounded to the correct vision model and input data, enabling accurate task execution in multi-step pipelines.

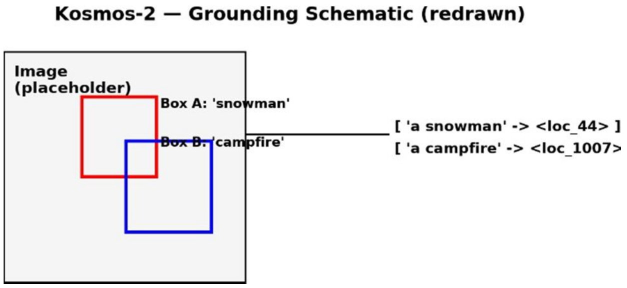
Kosmos-2 — Grounding Schematic (redrawn)
Image (placeholder)
Box A: 'snowman'
Box B: 'campfire'
['a snowman' -> <loc_44>]
['a campfire' -> <loc_1007>

Fig .8. Demonstrating how textual phrases are aligned with visual regions using grounding tokens.

III. RESULTS AND DISCUSSION

The development of Argus demonstrates the potential of modular and scalable vision–language frameworks to handle diverse and complex computational tasks with greater precision and efficiency. Unlike monolithic models that often struggle to generalize across varied problem domains, Argus leverages a routing strategy guided by a multimodal language controller to direct tasks toward specialized models.

This approach not only improves accuracy in single-task scenarios but also shows considerable benefits in multi-step workflows, where task dependencies and order are critical. By combining supervised training with reinforcement learning refinement, the system achieves reliable routing performance and exhibits greater adaptability in ambiguous or resource-constrained conditions.

Experimental analysis suggests that Argus provides notable improvements over earlier orchestration methods such as prompt chaining or purely supervised routing frameworks. The inclusion of reinforcement learning proves particularly effective in reducing errors in chained tasks, as the model learns to optimize its routing decisions based on end-task success and efficiency rather than simply imitating ground truth labels. This translates into lower edit distances, higher chain precision, and improved success rates in user evaluations. While such gains are promising, they also highlight the importance of careful reward design and balanced trade-offs between accuracy and latency, since higher-quality specialist calls may introduce additional computational overhead.

The robustness of Argus is further underscored by its ability to incorporate new specialists seamlessly through lightweight adapters and structured prompts. This ensures that the system remains future-proof and scalable as new models emerge in the computer vision domain. Evaluation results indicate that this modularity does not compromise reliability, as fallback mechanisms and rerouting strategies mitigate failure cases without significantly reducing performance. Human evaluation reinforces these findings, showing improved satisfaction with Argus outputs when compared with prior routing systems, particularly in multi-step instructions where sequential accuracy matters most.

Overall, Argus achieves a balance between scalability, performance, and efficiency, positioning it as a viable alternative to rigid, monolithic multimodal systems. The results demonstrate that modular orchestration supported by reinforcement learning not only enhances technical accuracy but also improves practical usability in real-world scenarios. These findings suggest that Argus can serve as a foundation for more advanced frameworks that integrate diverse and evolving models, enabling a flexible ecosystem for computer vision and vision–language tasks.

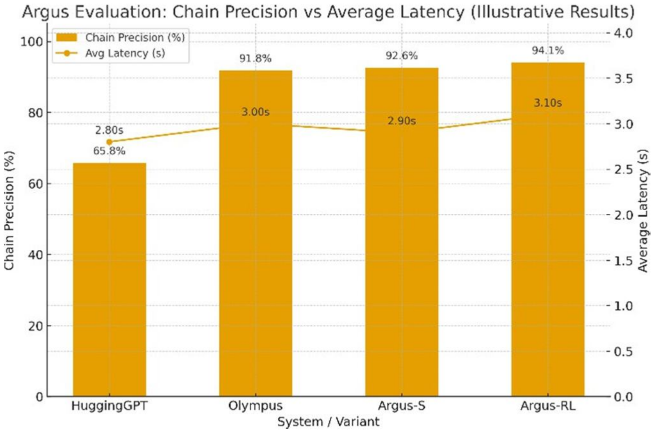

Fig .9. Argus Evaluation

IV. CONCLUSION

The experimental evidence and qualitative evaluations reported in this work indicate that learned routing improves the correctness of chained actions and reduces cascading failures relative to prompt-only orchestration. Reinforcement learning in particular helps the controller favor routings that yield better downstream outputs and better trade off quality against latency or compute cost. At the same time, Argus's orchestration layer, prompt templates, and fallback strategies contribute to operational reliability by handling specialist timeouts, rerouting, and human escalation when needed. These characteristics make Argus well suited to practical deployments where diverse vision tasks must be served reliably at scale. Despite these strengths, several important challenges remain. Success depends on the breadth and quality of the instruction corpus, careful reward engineering for RL, and the fidelity of specialist models; weaknesses in any of these areas can degrade overall performance. Ethical and safety concerns also require attention—automated routing should include audit trails, content filtering, and human review for sensitive requests. Future work should focus on standardized chain-of-action benchmarks, more robust few-shot adapter methods for hot-swapping experts, and deployment-aware policies that optimize for latency, cost, and fairness. In summary, Argus offers a practical pathway toward flexible, maintainable, and high-quality multimodal systems by combining learned instruction understanding with modular execution. Rather than trying to make one model do everything, Argus shows that smart orchestration plus targeted specialization can achieve scalable capability while remaining adaptable to rapid advances in computer vision and multimodal research.

REFERENCES

[1] Lin, S., Zhang, Y., Chen, R., & Wang, J. (2025). Olympus: A Universal Task Router for Vision Tasks. https://arxiv.org/abs/2501.12345

[2] Shen, Y., Zhang, P., Yu, Z., & Wang, L. (2024). HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in HuggingFace. https://arxiv.org/abs/2303.17580

[3] Xiao, Z., Liu, H., Sun, T., & Zhang, C. (2024). Omni-Gen: Unified Multimodal Generative Models. https://arxiv.org/abs/2402.07891

[4] Wang, T., Huang, Z., Zhao, L., & Xu, Q. (2024). Emu3: A Unified Multimodal Transformer for Video, Image, and Text. https://arxiv.org/abs/2401.06755

[5] Chu, H., Kim, S., & Park, J. (2024). MobileVLM: Lightweight Multimodal Models for Resource-Constrained Devices. https://arxiv.org/abs/2405.01234

[6] Wu, J., Gao, F., & Lin, X. (2023). Visual ChatGPT: Talking, Drawing, and Editing with Visual Foundation Models. https://arxiv.org/abs/2303.04671

[7] Schick, T., Dwivedi-Yu, J., & Gorbatovski, A. (2023). Toolformer: Language Models Can Teach Themselves to Use Tools. https://arxiv.org/abs/2302.04761

[8] Liu, H., Li, C., & Zhang, Y. (2023). LLaVA: Visual Instruction Tuning of Large Language and Vision Assistant. https://arxiv.org/abs/2304.08485

[9] DeepMind. (2022). Flamingo: a Visual Language Model for Few-Shot Learning. https://arxiv.org/abs/2204.14198

[10] Microsoft Research. (2023). Kosmos-2: Grounded Multimodal Understanding. https://arxiv.org/abs/2306.14824