GUI Agent for Device Control

We aim to develop an intelligent GUI control system that:

- Precisely Understands Mobile Interfaces
  - Combines visual screenshots + layout hierarchies
  - Extracts all interactive elements (buttons, switches, text fields) with pixel-accurate bounding boxes
- Automates Device Interactions
  - Classifies element interactivity
  - Generates correct action sequences (taps, swipes, text input)
  - Adapts to dynamic UI changes in real-time
- Generates Action Sequences
  - For each operation it takes, it can predict the next action.

FINAL GOAL: Create a system that can effectively interact with mobile devices

GOALS

SOTA Research Papers

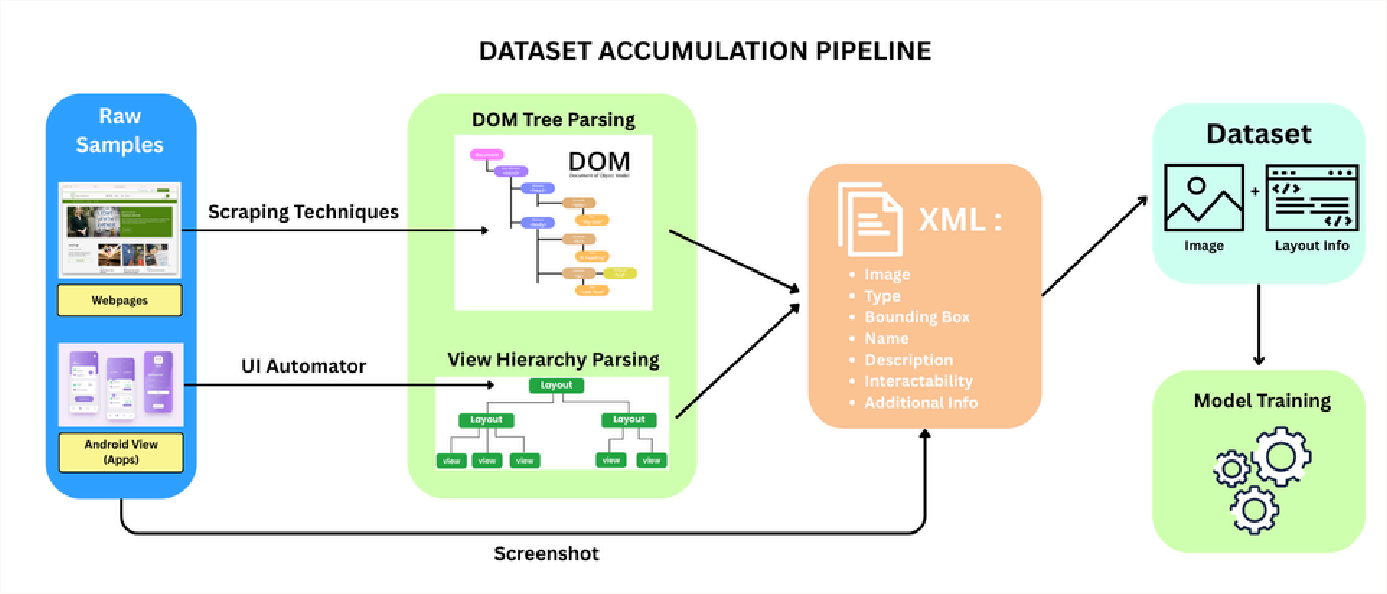
- Android in the Wild (AitW): Inspired by their multimodal transformer architecture combining VisionBERT and BERT embeddings, we adapt their approach to our GUI agent while focusing on real-time interaction rather than offline analysis.
- OmniParser: Drawing from their universal parsing framework, we implement similar cross-platform capabilities but specialize for mobile device control with enhanced interactivity detection.

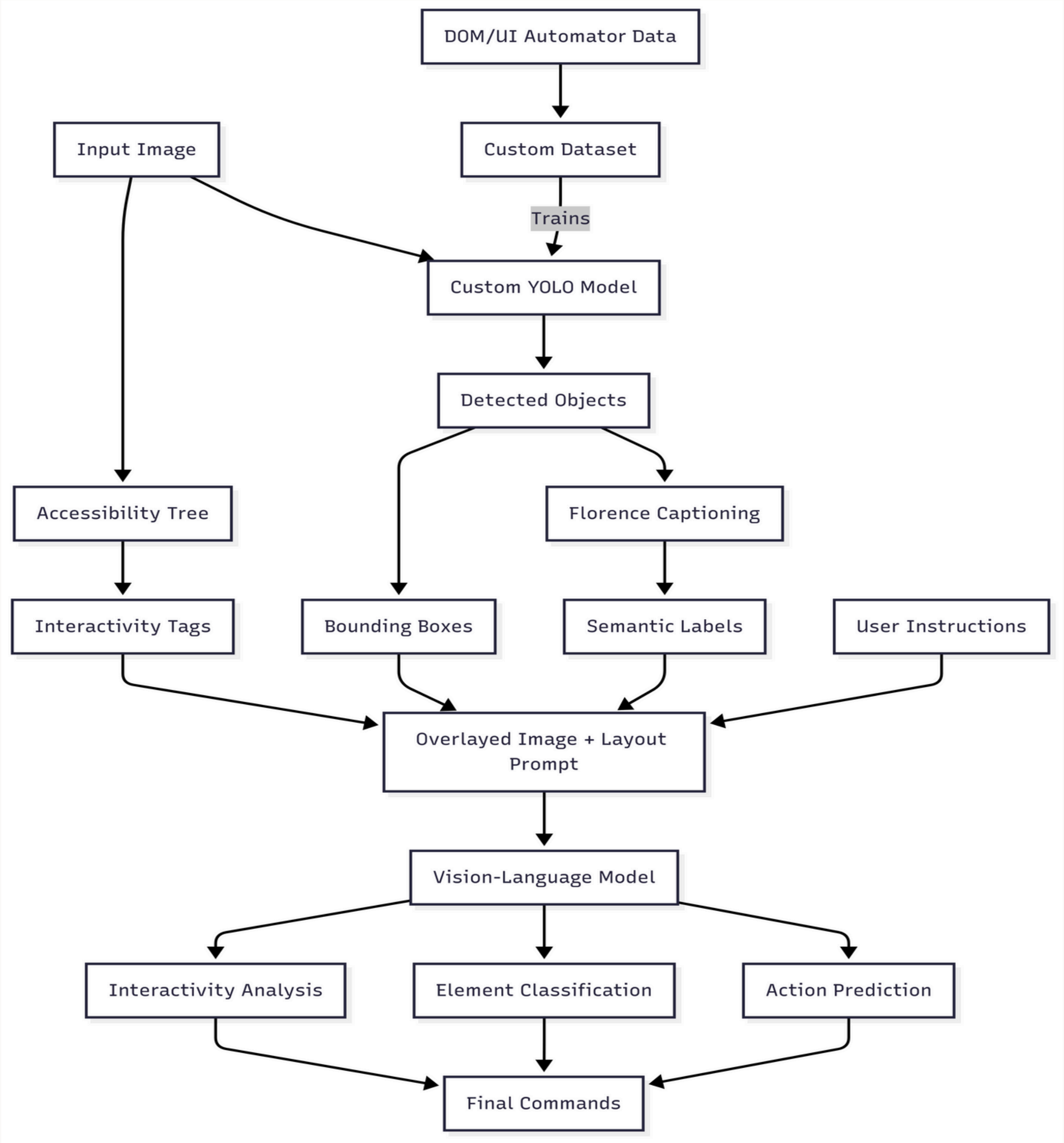
External References

- MobileViews – A dataset and model for UI element detection, focusing on view hierarchy and pixel-level segmentation to improve element localization.
- ScreenSpot – A vision-based approach for UI element retrieval using contrastive learning, enabling zero-shot detection of UI components without manual annotations.
- SeeClick – A reinforcement learning-based system for automating mobile interactions by predicting click coordinates and gestures from screenshots.

Understanding Layout Info

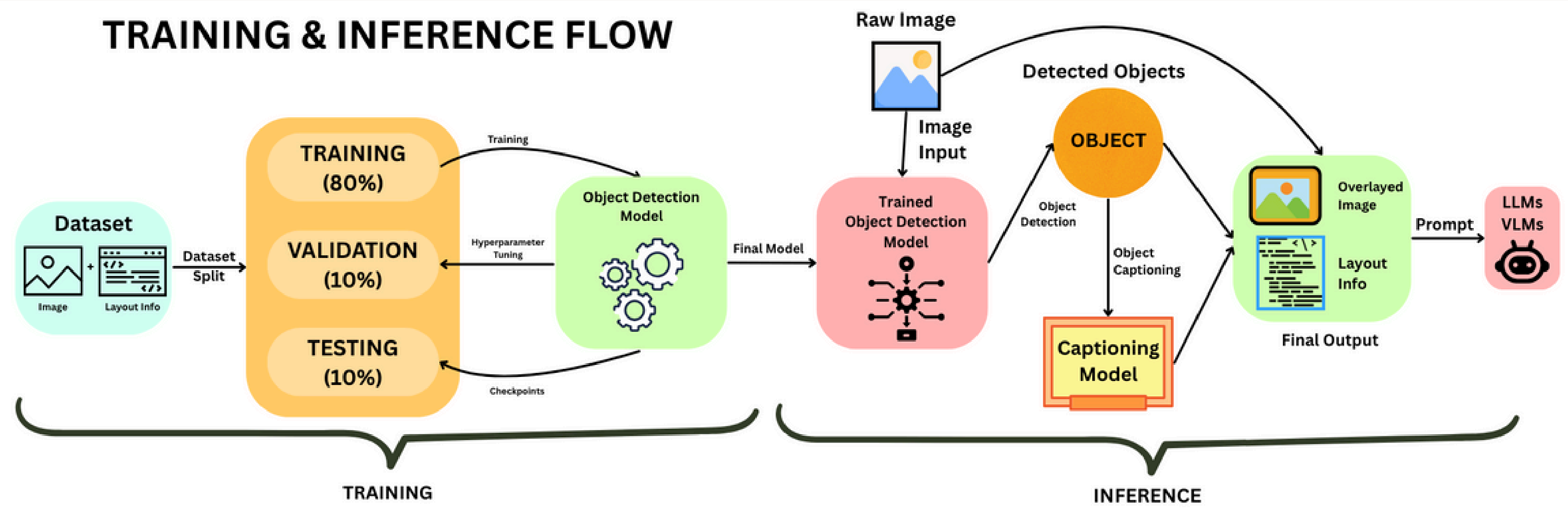
- Web UI Extraction:
  - Used DOM tree parsing to extract bounding boxes
  - Captured visible elements: icons, images, text, hyperlinks
- Mobile UI Extraction:
  - Implemented UI Automator framework
  - Collected same element types as web (icons, images, text, etc.)
- Accessibility Enhancement:
  - Built custom APKs to leverage Accessibility APIs
  - Acquired proper icon descriptions and accurate labels
- Interactivity Detection:
  - Developed binary classifier for clickability analysis
  - Generated interactivity labels for all elements
- OmniParser Pipeline Upgrade:
  - De-coupled YOLO model due to poor performance on custom dataset
  - Generate custom dataset for custom model training
  - Currently building specialized replacement model
  - Actively creating optimized training dataset combining:
    - Visual features
    - Semantic annotations
    - Cross-platform UI patterns



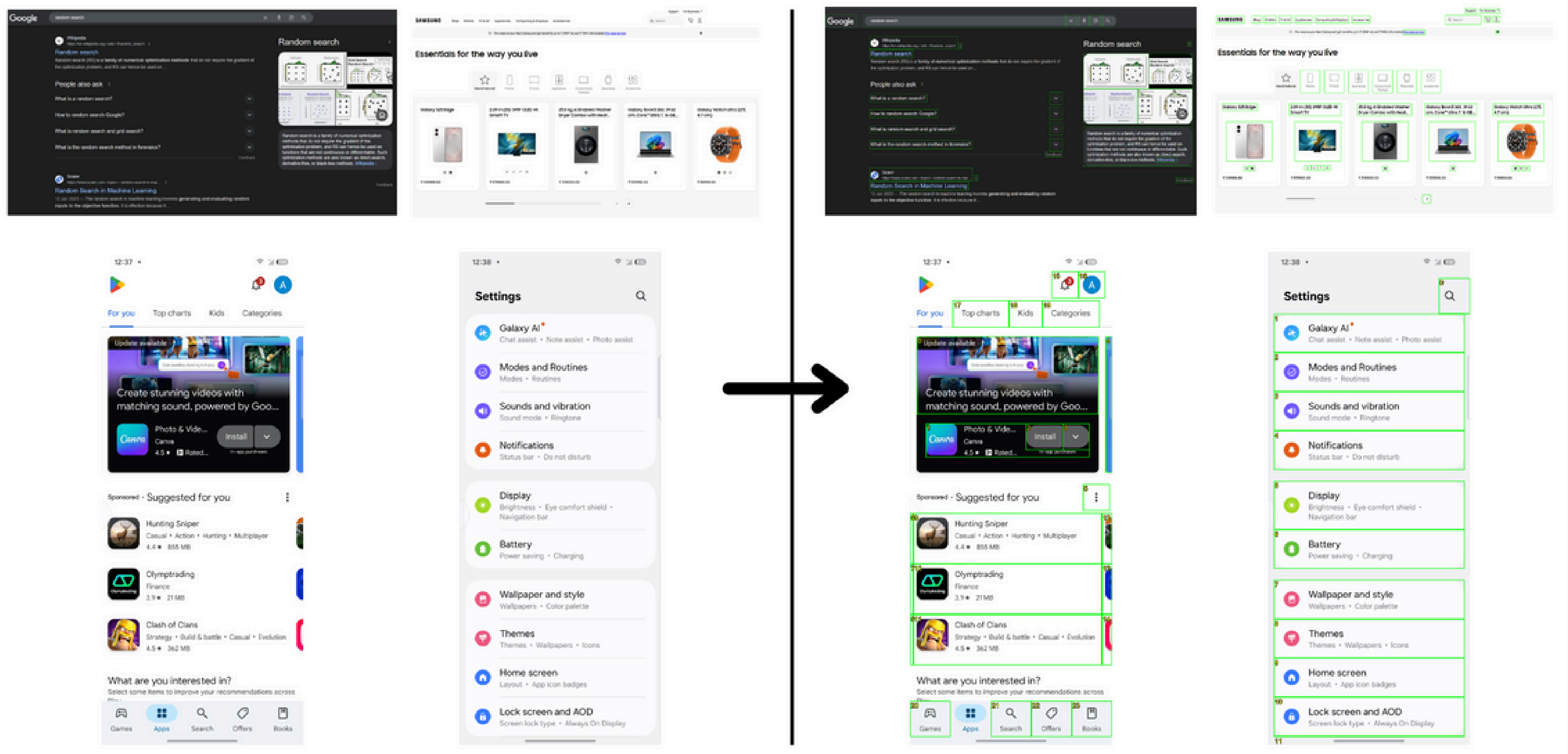


Current Status (Dataset Generation)

- Mobile UI Automation: Developed scripts using UI Automator with tree traversal approaches to capture screen snapshots and XML layout files, enabling structured parsing of mobile UI elements.
- Web UI Extraction: Built DOM tree parsing scripts for web automation, extracting bounding boxes, element types, and hierarchical relationships for comprehensive web interface analysis.
- Accessibility-Enhanced Labeling: Leveraged Accessibility APIs to generate precise object classifications (e.g., Image, Icon, Button) and improve labeling accuracy for training data.



Bounding Box Layout using DOM Tree Parsing & UI Automator



Key Features Implied:

✓ Cross-platform (Mobile + Web)

✓ Structured + visual data (XML + Screenshots)

✓ High-precision labels (Accessibility API corrections)

Completion Aim

✓ Research Paper Completion (Under Review & Updation)
Introduce novel hybrid technique:
• DOM Tree (Web) + UI Automator (Mobile)
• Accessibility Events (for interactivity labels)
Benchmarking against AitW & OmniParser (higher F1-score)

✓ Active Dataset Generation
Devices running custom APKs and Scripts
Auto-labeled 60K+ screens (target)

✓ Immediate Next Steps
ICVGIP’25 submission
Train interactivity-aware YOLOv11 variant:
Multi-task output (class + clickability probability)
On-device optimization for ms inference

Impact: First unified framework for visual + functional UI understanding.