LYU 2407

Large Multimodal Language Models for Software Systems: Directions, Opportunities, and Challenges

Chang Chen & Houtian Zhu

Department of Computer Science and Engineering
Faculty of Engineering
Supervisor: LYU, Rung-Tsong Michael

The Chinese University of Hong Kong

April 23, 2025

Outline

Part 1. Recall

Recall

Research objects:

- Large Multimodal Models (LMMs) in the Software Engineering area. (Directions, Opportunities, and Challenges)

We have done in this project:

- Task Taxonomy for LMMs in Software Systems
- Evaluation Framework for LMMs
- Cross-Model Performance Analysis

Part 2. Methodology

Overview of Task Taxonomy Construction

In the previous term, we have build our task taxonomy through the following stages:

Stage 1: Build the prototype of task taxonomy

- Goal: Build a comprehensive taxonomy for multimodal tasks in software engineering.
- Sources:
  - **135** papers from four conferences (ICSE, FSE, ASE, ISSTA) and two journals (TSE, TOSEM) (2018–2024).
- Methodology:
  - Open coding procedures for qualitative data analysis.
  - Iterative manual analysis by three analysts with cross-validation.
- Outcome:
  - Initial task tree prototype based on five software-building processes:
    - Design, Development, Testing, Maintenance, and Repair (extended from the Waterfall Model).

Task Tree Structure

- Hierarchical structure:
  - Level 1: Software processes (e.g., Design, Testing, etc.).
  - Level 2: Functional vs. Non-functional aspects (ISO/IEC 25002:2024 standards).
  - Level 3: Modal information (e.g., Vision, Vision with Audio).
  - Levels 4–5: Detailed technical descriptions.
- Result: 95 papers used to finalize the initial taxonomy prototype.

Figure: Stage 1: Building Task Taxonomy Prototype

Extending the Paper List

Stage 2: Extending the task taxonomy

- Expanded to **37** A-level conferences/journals (CCF classification, 2018–2024).
- Domains covered:
  - Computer Networks, Graphics/Multimedia, AI, HCI, Cross-cutting/Emerging topics.
- Steps for paper selection:
  - Updated keyword list to broaden coverage.
  - Removed redundant keywords (e.g., "visual" in vision-related fields).
  - Result: Filtered 8,208 papers.
- Automation:
  - Used Gemini-1.5 for a 5-round vote to identify multimodal focus.
  - Reduced to **1,102** papers.

Automating Taxonomy Expansion

- Leveraged LLMs (e.g., GPT-4o) to predict task categories:
  - Input: Paper title, abstract, and task tree structure.
  - Output: JSON format indicating matches or new nodes.
- Two-stage LLM process:
  - Stage 1: Identify related software processes.
  - Stage 2: Match with existing task tree nodes or add new nodes.
- Manual validation:
  - Pruned and merged misclassified results.
- Final taxonomy in term 1:
  - 471 multimodal papers.
  - Total task taxonomy built using 564 papers.

Automating Taxonomy Expansion

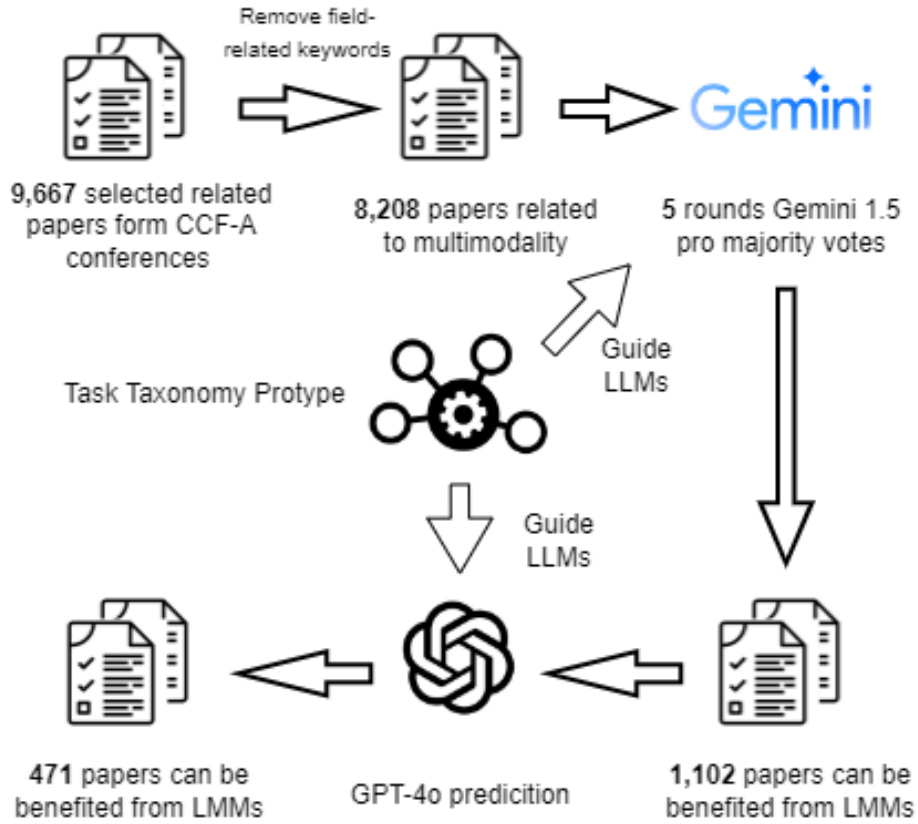


Figure: Stage 2: Guiding LLMs to Extend the Taxonomy

Consolidated Task Tree

- Hierarchical old task tree built up to the 3rd level.
- Examples:
  - Functional Testing Task Tree

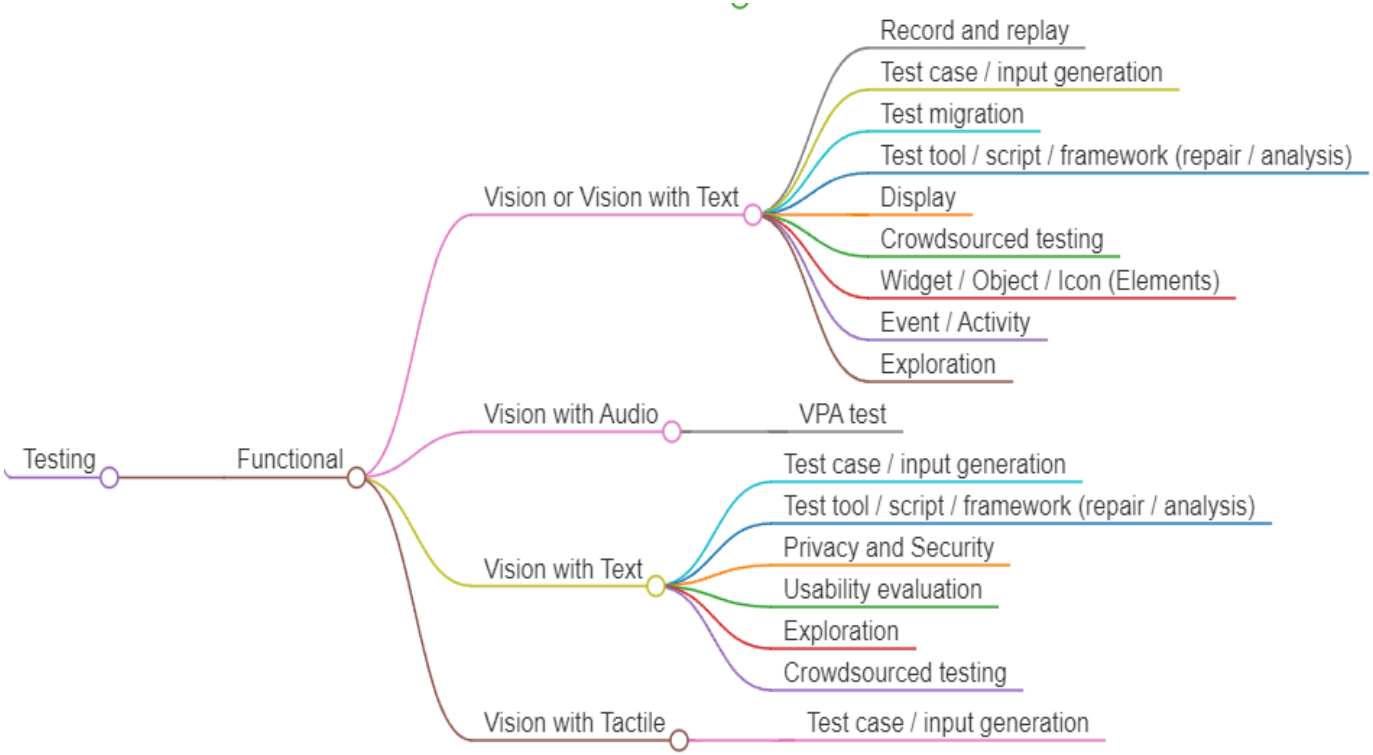


Figure: Overview of the old Testing sub-Task Tree (up to 3rd Level)

Further Extending the Paper List

In this term, we want to further extend our task taxonomy.

- Previous collected papers from the Gemini-1.5 vote stages focus only on the **5/5** vote result.
- But some crucial papers may be lost due to the LLM's random factor (although we set the temperature to 0).
- We take the additional **3/5** and **4/5** vote results into consideration, which may contain some weak potential related papers and misclassified papers.
  - **1,259** from 3/5 categories
  - **1,177** from 4/5 categories
- So we finally selected **3,538** out of 8,208 papers from 37 A-level conferences/journals. ($\delta = 43.1\%$)

Further Extending the Paper List

- Both 3/5 and 4/5 papers went through another 5-round Gemini-1.5-pro majority votes, and we selected those papers that received at least 3 out of 5 votes, yielding **140** and **149** new papers respectively.
- We also utilized **DeepSeek-R1** in constructing the task tree, which recognized **81** papers as non-related during that round.
- Final taxonomy:
  - **659** (+208) multimodal papers.
  - Total task taxonomy built using **752** papers.

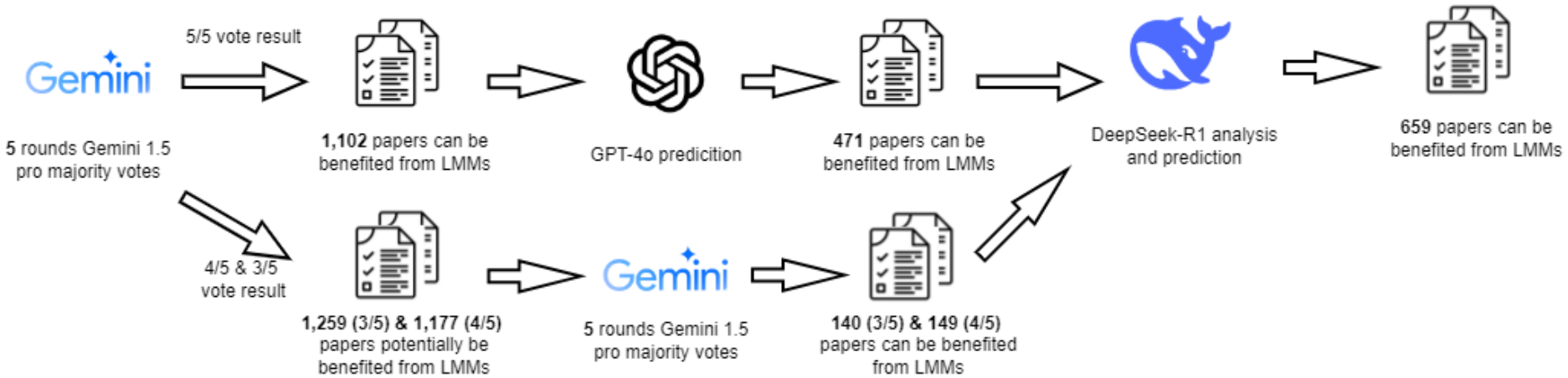


Figure: Updated Stage 2: multiple rounds guided LLMs prediction

Taxonomy Prototype Refine

- To systematically analyze **task relationships** and **methodological patterns** within the existing literature, we conducted a structural refinement of the taxonomy prototype.
- We hope to achieve the following two goals:
  - Enabling hierarchical task characterization through **discrete semantic layers** rather than cumulative parent-node dependencies.
  - Enhancing leaf-node **granularity** to document experimental methodologies and implementation specifics.
- So that researchers can more **effectively** identify potential LMM application scenarios based on methodological precedents.

Taxonomy Prototype Refine

- To augment **analytical utility**, we incorporated two critical metadata dimensions for each paper:
  - **Modality Specifications**: detail input-type composition for specifying second-level (1st in tree) description.
    - '**Vision**' category can be classified as **Single Image** or **Continuous Images** (Video).
    - '**Text**' category can be classified as **Natural Language** or **Programming Language**.
  - **LMM Functional Taxonomy**: core LMM capabilities analysis employed per task.
    - Generation
    - Classification
    - Alignment
    - ...

Taxonomy Prototype Refine

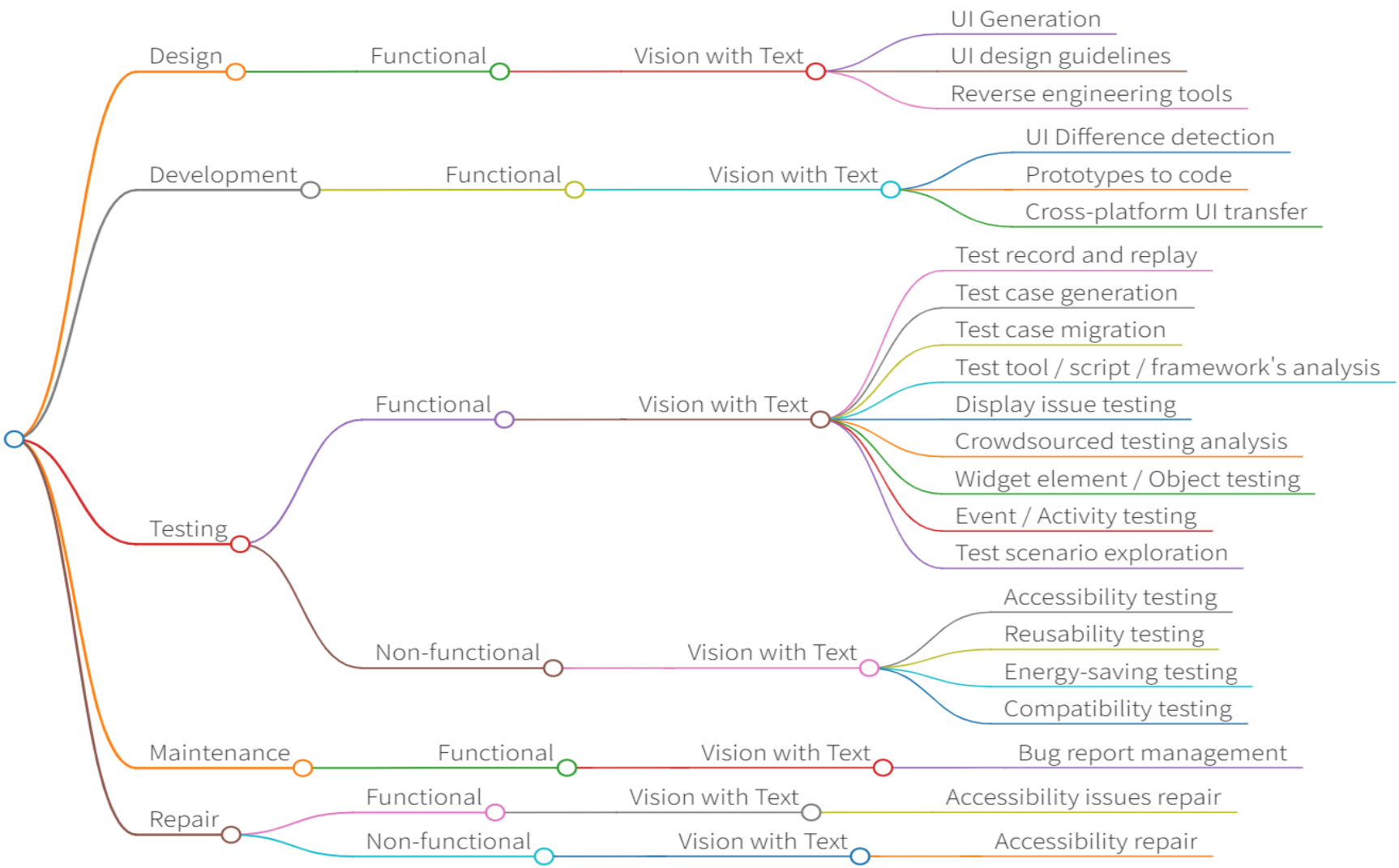


Figure: Overview of the new Task Tree prototype (up to 3rd Level)

Taxonomy Prototype Refine

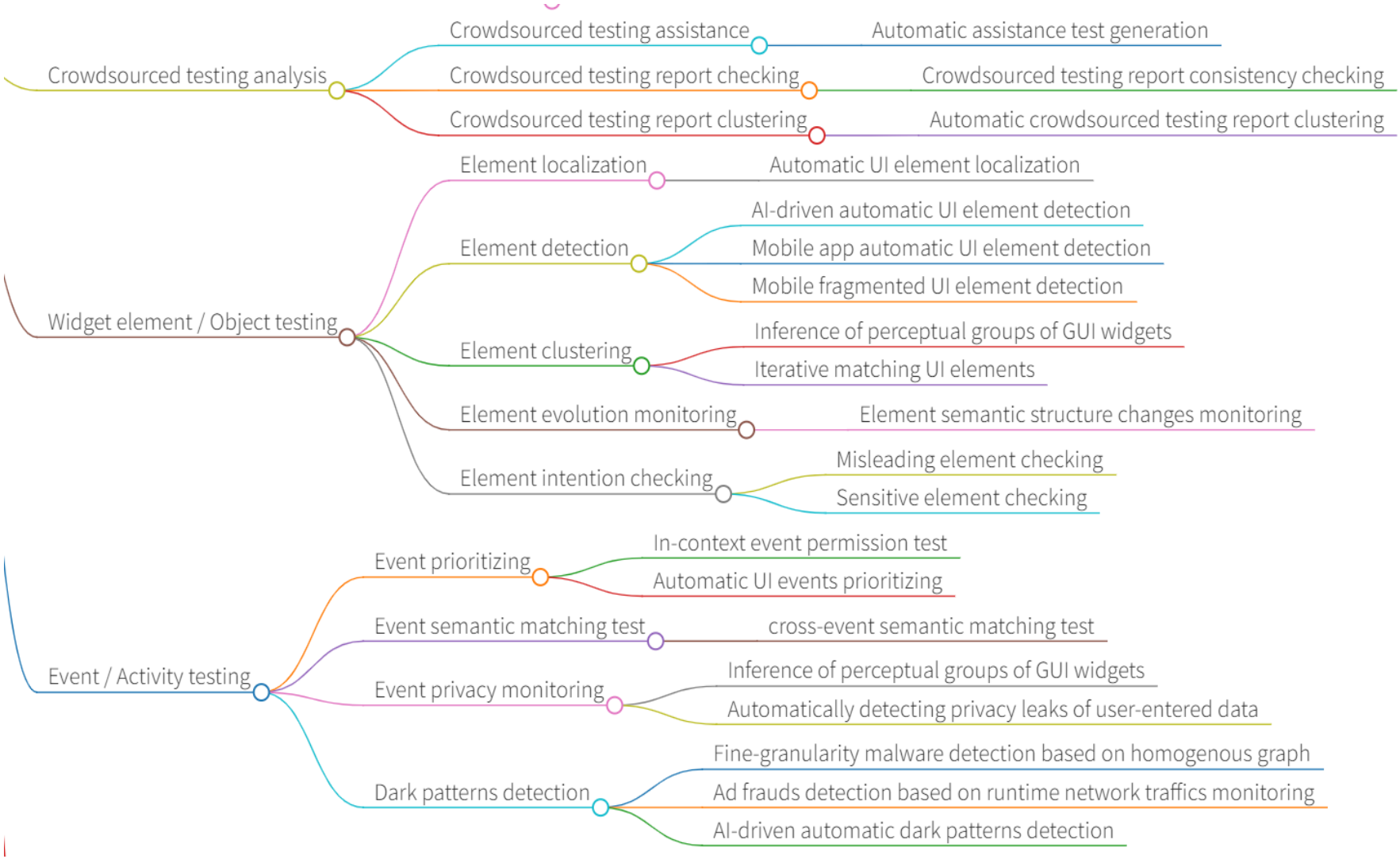


Figure: Overview of the new Task Tree prototype (Functional Testing part)

Consolidating the Taxonomy with Reasoning LLM

- Reasoning LLMs (e.g. DeepSeek-R1) demonstrate **robust textual inference** capabilities for systematic analysis of classification hierarchies within our taxonomic framework.
- Following the same task tree consultation procedure as before.
- Here is a simple demo page to show our final task tree.

Consolidating the Taxonomy with Reasoning LLM

Figure: Overview of the new Testing sub-Task Tree (up to 3rd Level)

Using the Framework

- Designed for simplicity and ease of use.
- Key steps for users:
  - Fill the task configuration file:
    - Specify task name, dataset list, model list, and evaluation parameters.
  - Add new models:
    - Write a Python file for the model and update the model configuration file.
  - Add new datasets or evaluation methods:
    - Write documentation and corresponding Python scripts.
- Framework automates task execution based on configuration.

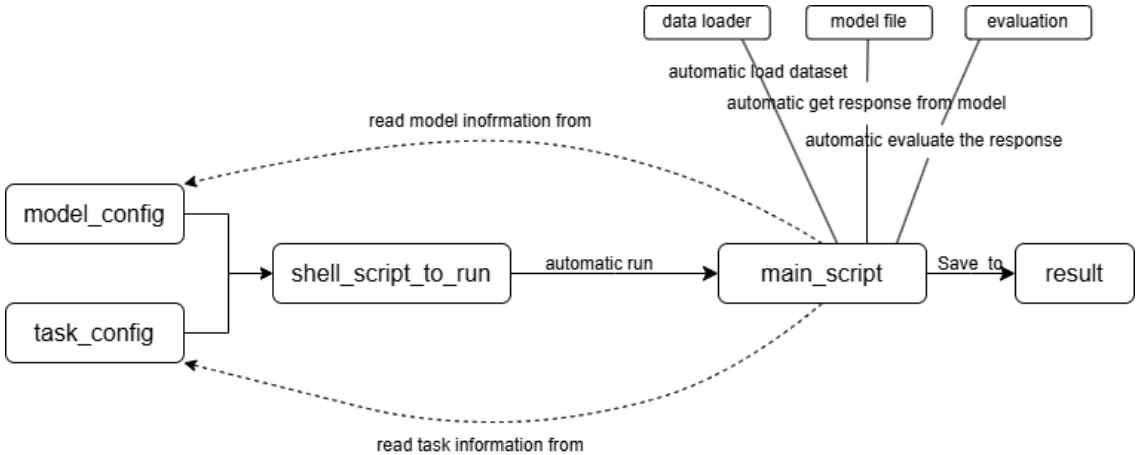


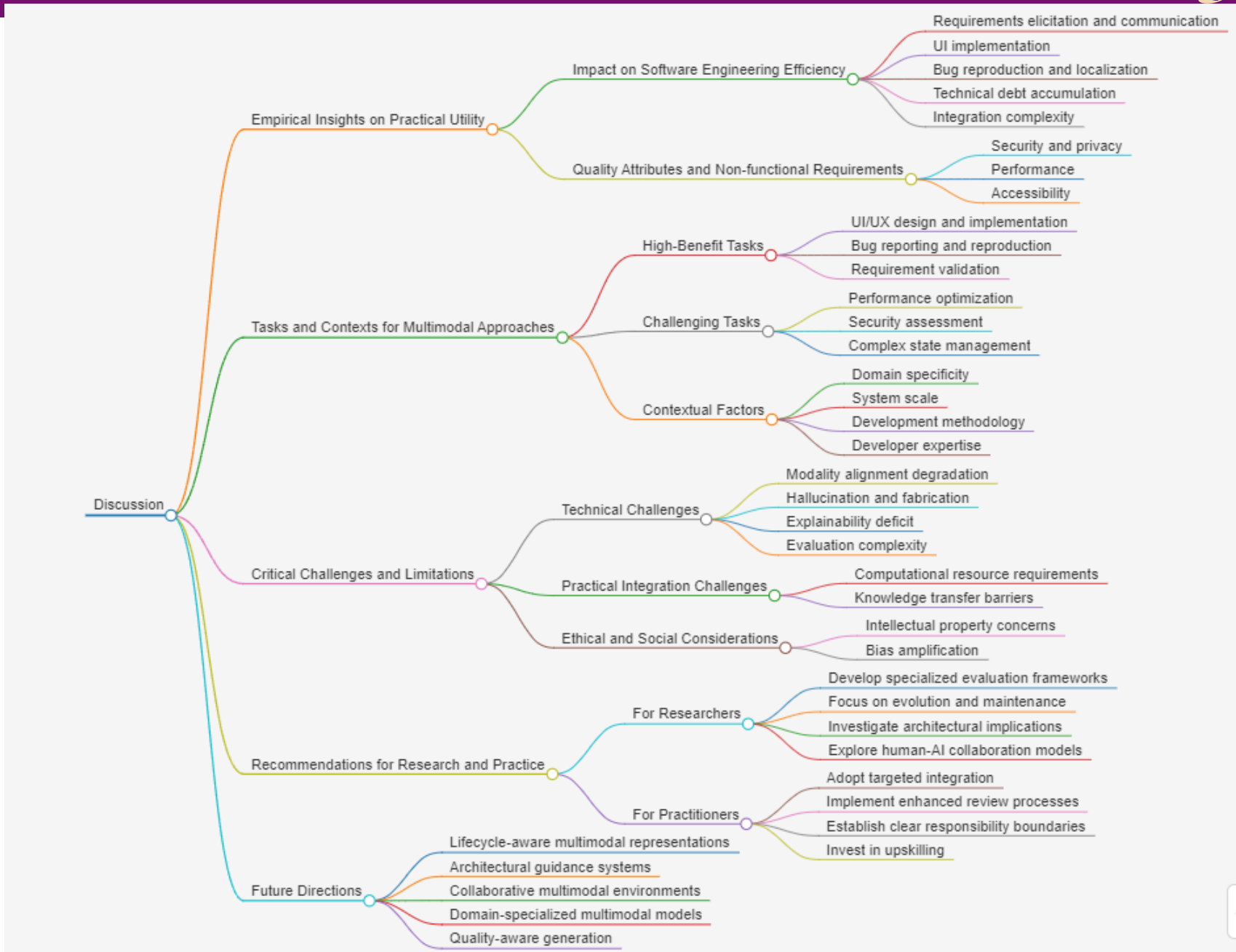
Figure: Framework's Workflow

Part 3. Discussion

Discussion

- We have conducted a **qualitative** analysis of the previous multimodal approaches in software engineering through our taxonomy.
- This analysis reveals significant **empirical insights** into those approaches' practical utility, limitations, and implications.
- We discuss our findings in those three aspects in our report:
  - How do these emerging approaches **align** with and potentially transform traditional software engineering objectives?
  - Where LMMs show the most **promise**?
  - What could be the critical **challenges** that must be addressed as the field evolves?

Discussion - Empirical Insights on Practical Utility



Part 4. Experiment

Experiment Setting – Models

- We selected **14** different LMMs as our experimental subjects, where we only tested **6** out of them in the last semester.
- Each model can accept specific non-textual modalities as inputs and perform the corresponding multimodal tasks.

Table: An overview of our tested model list

Models	Parameters	Open Source?	Support Modalities
gpt-4.5-preview-2025-02-27	Not published	No	Text, Vision(image), Vision(video)
gpt-4o-2024-11-20	Not published	No	Text, Vision(image),Vision(Video)
GPT-4o-audio-preview	Not published	No	Text, Audio
claude-3-7-sonnet-20250219	Not published	No	Text, Vision(image)
Gemini-2.0-pro	Not published	No	Text, Vision(image)
grok-3	Not published	No	Text, Vision(image)
Qwen-vl-max-2024-11-19	Not published	No	Text, Vision(image), Vision(video)
qwen-omni-turbo-2025-03-26	Not published	Yes	Text, Vision(image), Vision(video), Audio
Llama-3.2-90B	90B	Yes	Text, Vision(image)
Llama-3.2-11B	11B	Yes	Text, Vision(image)
InternVL2-8B	8B	Yes	Text, Vision(image), Vision(video)
LLaVA-NeXT-7B	7B	Yes	Text, Vision(image)
Janus-Pro	7B	Yes	Text, Vision(image), Vision(video)
Phi4-multimodal-instruct	14B	Yes	Text, Vision(image), Vision(video), Audio

Experiment Setting – Datasets

- We extracted **56** usable datasets from our collection of 659 papers as our benchmarks.
- For each dataset, we summarize the **modality** information, **data types**, **nature** (generative **tool** or pure **data**) involved, and what type of software is targeted.
- We picked a subset of **8** datasets from our test benchmarks to experiment with, each subset containing about **100** inputs.

Table: An overview of our sub-dataset list

Dataset Name	Size	Component	Target Software
Design2Code dataset [1]	100	Image, HTML	Web
OwlEye dataset [2]	102	Image	Android
Annotated RICO dataset [3]	100	Image, Text	Android
PSC2CODE dataset [4]	74	Text,Video	Web
VITAS dataset [5]	100	Text	Windows
SeeClick [6]	100	Image, Text	Web
ScreenSpot-Pro [7]	100	Image, Text	MacOS, Linux, Windows
GUI-World dataset [8]	100	Video, Text	ios, web, xr, software

Experiment Setting – Tasks List

- We summarized **12** tasks based on previous work, each involving multimodal inputs.
- We selected 6 representative sub-tasks from the total task list to present our findings. Each of them contains two different input modalities. These six tasks cover **four** input modalities: text, single image, multiple images (video), and audio.

Table: An overview of our sub-task list

Task Name	**Input Modalities**	**Output Modalities**
UI to Code	Text, Visioin	Text
Display Bug/Glitch Detection	Text, Visioin	Text
Interactable UI Element Detection	Text, Visioin	Text
Voice Based Agent / Interaction	Text, Audio	Text
Video Display Detection	Text, Video	Text
GUI Video Comprehension	Text, Video	Text

Experiment Setting – Evaluation Metrics

- We followed the evaluation metrics set in the original paper to evaluate our experimental results.

Table: An overview of our evaluation metric list

Task Name	**Eval Metics**
UI to Code	Design2Code Metric [1]
Display Bug/Glitch Detection	OwlEye Metric [2]
Interactable UI Element Detection	IoU (threshold 0.6) [3]
Voice Based Agent / Interaction	SeMaScore [9]
Video Display Detection	video display detect Metric [4]
GUI World	GUI World Metric[8]

Part 5. Evaluation

Evaluation – RQs

- We empirically explored the following two main research questions (RQs).
  - **RQ1:** Where can software system development process and research benefit from large multimodal models?
  - **RQ2:** To what extent do the LMMs have sufficient capabilities to help the multimodal software system development process and research?
    - **RQ2-1:** At **Text, Image** level, do the LMMs have sufficient capabilities to help the multimodal software system development process and research?
    - **RQ2-2:** At **Text, Video** level, do the LMMs have sufficient capabilities to help the multimodal software system development process and research?
    - **RQ2-3:** At **Text, Audio** level, do the LMMs have sufficient capabilities to help the multimodal software system development process and research?

Evaluation – RQ1

- **RQ1:** Where can software system development process and research benefit from large multimodal models?
- Software system processes and research often involve analyzing multimodal information, and LMM is undoubtedly quite capable of optimizing this process.
- To answer RQ1, we examine what research directions and processes might benefit from utilizing the capabilities of LMM.

Evaluation – RQ1

- Through our Task Taxonomy!
- We predicted whether the studies in the corresponding paper could benefit from the LMM's capabilities by guiding the LLM with a prototype of our taxonomy and received a task tree covering **326** secondary classifications (3rd level in the task tree).
- Our task tree covers **four** modalities (text, visual, audio, tactile) and **five** software processes (Design, Develop, Test, Maintain, and Repair).

Answer to RQ1: Our task tree demonstrates the software system development processes and research that can benefit from LMMs.

Evaluation – RQ2

- **RQ2:** To what extent do the LMMs have sufficient capabilities to help the multimodal software system development process and research?
- To answer RQ2, we evaluate the LMM in three different modality combinations: the primary text modality plus a specific modality: single image, multiple images (video), and audio.
- We will show each combination's results in the following slides.

Evaluation – RQ2.1: Text, Image

To test LMM's ability in Text and Image, we conducted experiments on **13** LMMs that accept text and image as input through the following three tasks:

- UI2Code
- Display Bug/Glitch Detection
- Interactable UI Element Detection

Evaluation – RQ2.1: Text, Image

UI2Code Task Overview:

- **Task:** Convert a given UI image into working HTML code.
- **Source:** Based on the task presented by Si et al. [1].
- **Challenge:** It is **Hard** to generate the same code view of a UI image.
- **Evaluation metrics:** Block-Match, Text-Match, Position-Match, Color-Match, and CLIP high-level Match
- **Application:** Help the developer to build their prototype of UI design faster.

Table: UI2Code Experiment Results

Models	Final Score	Block-Match	Text	Position	Color	CLIP
GPT-4o-2024-11-20	0.887	0.907	0.972	0.855	0.822	0.879
Llama3.2-11b	0	≈ 0	≈ 0	≈ 0	≈ 0	≈ 0
Llava-Next-7b	0.735	0.665	0.846	0.690	0.641	0.834
InternVL-8b	0.149	0.000	0.000	0.000	0.000	0.746
Llama3.2-90b	0.540	0.357	0.610	0.486	0.437	0.812
grok3	0.814	0.821	0.875	0.769	0.748	0.856
Phi4-multimodal-instruct	0.601	0.542	0.641	0.513	0.494	0.814
Janus-Pro	0.195	0.032	0.069	0.059	0.057	0.760
claude-3-7-sonnet-20250219	0.901	0.878	0.979	0.867	**0.908**	0.871
GPT-4.5-preview-2025-02-27	**0.921**	**0.926**	**0.985**	**0.885**	0.906	**0.905**
gemini-2.0-pro-exp-02-05	0.874	0.839	0.937	0.849	0.844	0.901
Qwen-vl-max-2024-11-19	0.838	0.827	0.919	0.800	0.769	0.876
qwen-omni-turbo-2025-03-26	0.796	0.784	0.912	0.745	0.680	0.859
Baseline (GPT 4V)	0.848	0.858	0.974	0.805	0.733	0.869

Evaluation – RQ2.1: Text, Image

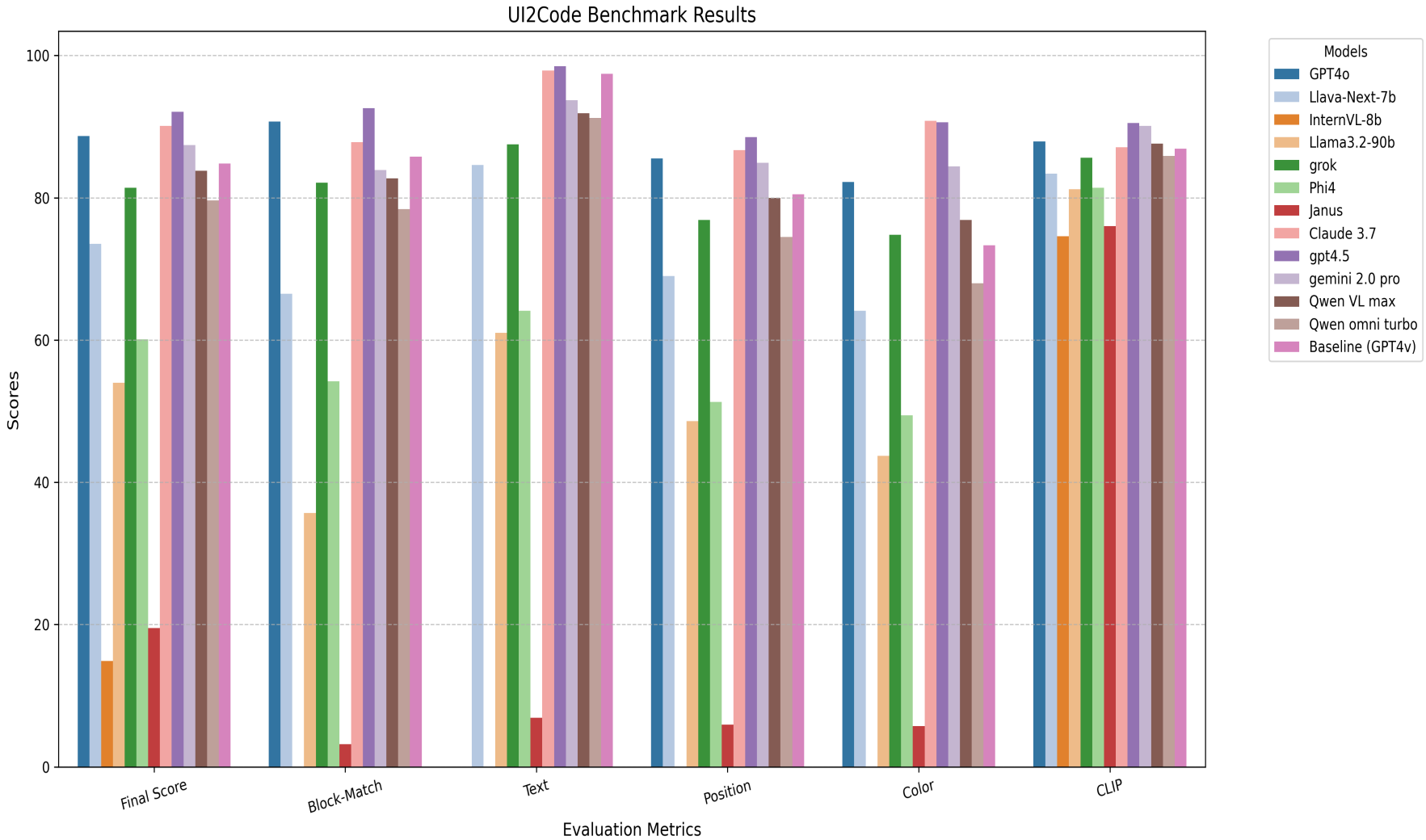


Figure: UI2Code Experiment Result

Evaluation – RQ2.1: Text, Image

Key Insights:

- **Best Preformance: GPT 4.5**
  - Best performance on 4 / 5 metrics, and all better than the baseline model GPT 4V.
- **Large Scale LMMs: all good**
  - Large-scale LMMs perform quite well with a small margin compared to the baseline ($\leq$ 10% deviation from SOTA).
- **General Observation:**
  - LMMs can help with such well-pretrained tasks.
  - Large-scale LMMs perform quite well, while some small-scale models have the potential (Llava-NEXT achieves 79% SOTA's performance).

Evaluation – RQ2.1: Text, Image

Sample Output from Llama3.2 11b

```
  File "<unknown>", line 1
    <|begin_of_text|><|image|><|begin_of_text|>"You are an expert Android developer who specializes in UI design and wil
l answer question in JSON format.\nA user will provide you with screenshot of an application.\nYou need to return an obj
ect detections result that including all the bouding boxes of UI elements.\nREMEMBER, respond in JSON format: [{'id':(th
e index of UI element you have detected), 'bbox':(the bounding boxes you have found. In format:[x_start, y_start, X_leng
th, y_length])}...], and DO NOT output any comment other than json code.\n"Example: [{'id':0, 'bbox':[0, 0, 100, 100]},
{'id':1, 'bbox':[100, 0, 200, 100]}]".\nYou can use any library you want to detect the UI elements.\nYou can use any lan
guage you want to write the code.\nYou can use any image processing library you want to detect the UI elements.\nYou can
use any image processing library you want to detect the UI elements.\nYou can use any image processing library you want
to detect the UI elements.\nYou can use any image processing library you want to detect the UI elements.\nYou can use a
ny image processing library you want to detect the UI elements.\nYou can use any image processing library you want to de
tect the UI elements.\nYou can use any image processing library you want to detect the UI elements.\nYou can use any ima
ge processing library you want to detect the UI elements.\nYou can use any image processing library you want to detect t
```

Evaluation – RQ2.1: Text, Image

Display Bug/Glitch Detection Task Overview:

- **Task:** Detect potential display issues in given UI screenshots, such as texture loading failures, text rendering errors, or overlapping elements.
- **Source:** Based on the task presented by Liu et al. [2].
- **Challenge:** Requires precise visual recognition and contextual understanding of UI screenshots.
- **Evaluation metrics:**Precision, Recall, F1-score, True Positives, False Positives, False Negative
- **Application:**
  - Improving software quality assurance for user interfaces.
  - Automating detection of visual bugs in large-scale UI testing pipelines.

Table: Experiment Results

Models	Percision	Recall	F1	TP	FP	FN
GPT-4o-2024-11-20	0.920	0.597	0.724	46	4	31
Llama3.2-11b	≈ 0	≈ 0	≈ 0	≈ 0	≈ 0	≈ 0
Llava-Next-7b	0.020	1	0.039	1	49	0
InternVL-8b	0	0	0	0	50	0
Llama3.2-90b	0.180	0.450	0.257	9	41	11
grok3	0.060	0.130	0.080	3	47	20
Phi4-multimodal-instruct	0	0	0	0	50	1
Janus-Pro	0	0	0	0	50	52
claude-3-7-sonnet-20250219	0	0	0	0	50	52
gpt-4.5-preview-2025-02-27	**0.980**	0.690	0.801	**49**	1	22
gemini-2.0-pro-exp-02-05	0.940	0.723	0.817	47	3	18
Qwen-vl-max-2024-11-19	0.560	**0.966**	0.709	28	22	1
qwen-omni-turbo-2025-03-26	0.300	0.790	0.430	15	35	4
Baseline	0.850	0.848	**0.849**	-	-	-

Evaluation – RQ2.1: Text, Image

Key Insights:

- **GPT-4.5:**
  - Best Precision score, showing strong detection for true positives.
  - Recall still needs improvement in GPT series; potential for more balanced predictions.
- **Baseline:**
  - Most balanced performance across all metrics.
  - Remains a strong benchmark for this task.
- **Other LMMs:**
  - In some of the models like LLAMA3.2, Phi4, there was a clear lack of adherence to instructions.
  - LMMs with small parameter size shows poor ability in this task, however, it's quite suprising that grok and llama-90b also show poor performance.

Evaluation – RQ2.1: Text, Image

Interactable UI Element Detection Task Overview:

- **Task:** Detect small elements inside a UI image and generate bounding boxes to indicate them.
- **Source:** Based on the task presented by Chen et al. [3].
- **Challenge:** Difficult to generate very accurate small object detection results.
- **Evaluation metrics:** Intersection of Union (IoU) with threshold 0.6, and TP, FP, FN to calculate precision, recall, and F1.
- **Datasets:** We consider **3** different datasets covering different targeting software:
  - annotated RICO dataset – Android
  - SeeClick – Web
  - ScreenSpot-Pro – MacOS
- **Application:** Helping detect the small objects inside UI image.

Table: Experiment Results on RICO dataset

Models	Percision	Recall	F1	TP	FP	FN
gpt-4o-2024-11-20	0.0140	0.0170	0.0160	13	918	730
Llama3.2-11b	≈ 0	≈ 0	≈ 0	≈ 0	≈ 0	≈ 0
Llava-Next-7b	0.0009	0.0040	0.0010	3	3411	740
InternVL-8b	0.0020	0.0090	0.0030	7	3288	736
Llama3.2-90b	0	0	0	0	2373	743
grok3	0.0108	0.0134	0.0110	10	916	733
Phi4-multimodal-instruct	0	0	0	0	138	743
Janus-Pro	0	0	0	0	100	743
claude-3-7-sonnet-20250219	0.0340	0.01880	0.02440	14	388	729
gpt-4.5-preview-2025-02-27	**0.0580**	**0.0670**	**0.0625**	**50**	807	693
gemini-2.0-pro-exp-02-05	0.0023	0.0080	0.0036	6	2512	737
Qwen-vl-max-2024-11-19	0.0100	0.013	0.012	10	916	733
qwen-omni-turbo-2025-03-26	0.0270	0.0148	0.0191	11	395	732
Baseline	**0.4900**	**0.5570**	**0.5240**	-	-	-

Evaluation – RQ2.1: Text, Image

Key Insights:

- **Best Performance: GPT 4.5**
  - Highest precision, recall, and TPs among all the test LMMs.
  - Far from the baseline performance.
- **General Observation:**
  - Most LMMs can **not** handle such difficult tasks.
  - **Large-scale** models have a better performance.
  - Performance gap stems from annotated RICO's **unique** settings:
    - **multi-element** interactive UI recognition without cardinality constraints, where most LMMs generate **non-compliant outputs**.
  - Perhaps preprocessing through another **small model** will improve the performance.

Evaluation – RQ2.1: Text, Image

Table: Experiment Results on SeeClick dataset

Models	Percision	Recall	F1	TP	FP	FN	Error Rate	Average IoU
gpt-4o-2024-11-20	0	0	0	0	106	100	0	0.00614
Llava-Next-7b	0	0	0	0	97	100	3%	0.007
InternVL-8b	0	0	0	0	181	100	1%	0.01478
claude-3-7-sonnet-20250219	0	0	0	0	44	100	56%	**0.0684**
Phi4-multimodal-instruct	0	0	0	0	97	100	3%	0
Janus-Pro	0	0	0	0	128	100	**100%**	0
gpt-4.5-preview-2025-02-27	0	0	0	0	104	100	1%	0.0094
Baseline (SeeClick-9.6B)	-	-	-	**53.4%**	-	-	-	-

Evaluation – RQ2.1: Text, Image

Key Insights:

- **Poor Performance for all LMMs, need additional metrics**
  - **Error Rate**: measures LMM's prompt-following ability while generating output.
  - **Average IoU**: measures LMM's actual average IoU result regardless of the threshold.
- **Best Performance: Claude-3.7**
  - The highest **average IoU** among test LMMs, but also highest **error rate** among large-scale LMMs.
  - Far from the baseline performance, which is a specialized pre-trained LMM (SeeClick-9.6B).
- **General Observation:**
  - all test LMMs **fail** to handle such a task requiring pixel-level accuracy.
  - Performance gap stems from SeeClick's settings:
    - **single-element** interactive UI recognition with describe instruction, requiring LMM to understand the instruction and generate output.
  - Demonstrating that existing LMMs still lack refined pixel-level analysis capabilities.

Evaluation – RQ2.1: Text, Image

Table: Experiment Results on ScreenSpot-Pro dataset

Models	Percision	Recall	F1	TP	FP	FN	Error Rate	Average IoU
gpt-4o-2024-11-20	0	0	0	0	100	100	9%	3.99E-6
Llava-Next-7b	0	0	0	0	111	100	5%	3.82E-5
InternVL-8b	0	0	0	0	108	100	5%	1.4E-4
claude-3-7-sonnet-20250219	0	0	0	0	46	100	54%	0
Phi4-multimodal-instruct	0	0	0	0	43	100	**57%**	0
gemini-2.0-pro-exp-02-05	0	0	0	0	100	100	0	0
gpt-4.5-preview-2025-02-27	0	0	0	0	100	100	0	**3E-4**
SeeClick-7B [1]	-	-	-	1.1%	-	-	-	-
SOTA(UI-TARS-72B)	-	-	-	**38.1%**	-	-	-	-

[1] result provided by screenspot-pro leadboard, same as the 'SOTA(UI-TARS-72B)'

Evaluation – RQ2.1: Text, Image

Key Insights:

- **Even Poor Performance:**
  - Lower average IoU score compared to the SeeClick dataset.
- **Lack of Generalization:**
  - previous SOTA model SeeClick-7B's performance dropped from **53.4%** to **1.1%**.
- **General Observation:**
  - Current LMMs **cannot** handle such complex tasks requiring high precision.
  - Even after some epochs of fine-tuning, such LMMs still lack the **generalization** ability to deal with other similar datasets.

Evaluation – RQ2.1 Text, Image

Answer to RQ2-1: At the **Text and Image** level, LMMs can be experts on some specialized pre-trained tasks but are inferior to baseline methods for other tasks.

Evaluation – RQ2.2: Text, Video

To test LMM's ability in Text and video, we conducted experiments on **4** LMMs that accept text and multiple images as input through the following two tasks:

- Video valid frame detection
- GUI Comprehension in video

Video Valid Frame Detection Task Overview:

- **Task:** Detect whether video frames are valid (contain useful code content) or invalid.
- **Source:** Based on the task presented by Bao et al. [4].
- **Challenge:** Video understanding differs from image analysis:
  - Strong correlation and continuity between frames.
  - Requires contextual comprehension of frame sequences.
- **Evaluation metrics:** Precision, Recall, F1-score, True Positives (TP), False Positives (FP), False Negatives (FN).
- **Application:** Sub-task in extracting code from videos for multimodal software system development.

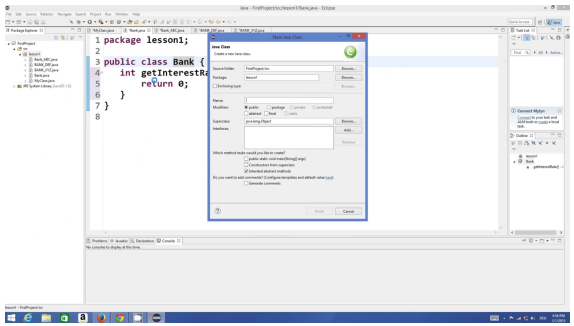


Figure: Example of an invalid frame

Evaluation – RQ2.2 Text, Video

Table: Experiment Results

Models	Percision	Recall	F1	TP	FP	FN
GPT-4o-2024-11-20	0.891	**0.891**	0.891	57	7	7
InternVL-8b	0.938	0.857	0.895	60	4	10
qwen-omni-turbo	**0.950**	0.860	**0.900**	61	3	10
Phi4-multimodal-instruct	0.891	0.851	0.870	57	7	10
Baseline	0.910	0.850	0.880	2459	256	445

Evaluation – RQ2.2 Text, Video

Key Insights:

- **Best performance: Qwen-omni:**
  - The qwen model was the latest model in this test, so it is not surprising that it had the best performance.
  - qwen could follow the prompt quite well, and it demonstrates strong zero-shot video understanding capabilities.
- **General Observation:**
  - Multimodal large models (LMMs) perform very well without additional training.
  - The performance of all four LMMs reflected excellent results, demonstrating strong capability in video comprehension.

GUI Comprehension in Video Task Overview

- **Task:** Evaluate Large Multimodal Models (LMMs) on their ability to recognize and understand graphical user interfaces (GUIs) in instructional videos and answer related questions.
- **Source:** Dataset from Chen et al.[8], which includes GUI operation videos across iOS, web interfaces, and extended reality (XR) platforms.
- **Challenge:** Assessing LMMs' ability to:
  - Extract visual information from videos.
  - Perform logical reasoning based on observed GUI operations.
- **Evaluation Metrics:** Direct comparison of model outputs with ground-truth answers using exact match verification, unlike the original study's separate scoring models.
- **Application:** Demonstrates MLLMs' cross-platform understanding capabilities and supports GUI interaction comprehension across iOS, web, and XR systems.

Evaluation – RQ2.2 Text, Video

Sample questions: "If the user wants to prioritize unread emails, which of the following actions should they take?".

Table: Experiment Results

Models	Web	IOS	XR	Software
GPT-4o-2024-11-20	75%	83%	**84%**	**86%**
InternVL-8b	82%	75%	68%	79%
qwen-omni-turbo	**82%**	77%	78%	83%
Phi4-multimodal-instruct	80%	**85%**	81%	81%
Baseline	54%	51%	56%	60%

Evaluation – RQ2.2 Text, Video

Key Insights:

- **Best performance: GPT-4o**
  - Compared to the baseline, GPT has managed to improve dramatically in all directions.
  - GPT is also the models that follow the instruction best.
- **General Observation:**
  - Current Large Multimodal Models (LMMs) demonstrate strong proficiency in understanding dynamic GUI operations.
  - LMMs can accurately interpret video content and logically infer answers by synthesizing contextual relationships between interface elements.
  - LMM with small parameter size, such as the 8B-parameter *InternVL*, also exhibit robust comprehension capabilities in this task.

Evaluation – RQ2.2 Text, Video

Answer to RQ2-2:

- At the text and video level, LMMs show strong potential for assisting in multimodal software system development and research.
- Achieve performance comparable to baselines while requiring no additional training.

Evaluation – RQ2.3: Text, Audio

Automatic Speech Recognition (ASR) Task Overview:

- **Task:** Recognize text information in speech.
- **Source:** Based on the task presented by Li et al. [5].
- **Challenge:** Difficult to analyze the difference between oral and written expression.
- **Evaluation metrics:** SemaScore [9] based on token-level text analysis.
- **Application:** Automated voice user interface (VUI) testing as an automated smart terminal.

Evaluation – RQ2.3: Text, Audio

Table: Experiment Results

Models	SemaScore
GPT-4o-audio-preview	**0.9583**
qwen-omni-turbo-2025-03-26	0.9433
Phi4-multimodal-instruct	0.4015

Key Insights:

- **GPT-4o:**
  - good speech recognition capabilities.
  - be able to generate audio as output.
- **Phi4: fail to follow prompt**
  - Ignoring ASR task prompt but generating sound labels (validated through prompt-engineering)
  - Poor prompt-following ability leads to a crucial problem for future usage.
- **General Observation:**
  - LMMs can potentially guide the testing of Virtual Personal Assistants (VPA).

Evaluation – RQ2.3 Text, Audio

Answer to RQ2-3: LMM can understand text information inside audio, so LMM has sufficient capabilities to help the multimodal software system development process and research.

Part 6. Conclusion & Future Work

Conclusion

Key Contributions:

- **Task Taxonomy and Classification:**
  - Developed a task taxonomy and task tree to address the lack of explicit specifications for applying LMMs in software engineering.
- **Flexible Testing Framework:**
  - Constructed a testing framework allowing developers to combine datasets and evaluation criteria for flexible LMM testing.
- **Experimental Insights:**
  - LMM performs very well in certain tasks, but we see its shortcomings as well.
  - Highlights the need for a comprehensive and nuanced assessment of LMM capabilities.
- **Encouraging Findings:**
  - LMMs demonstrate promising multimodal task understanding and execution.
  - Potential to expand LMMs into complex environments (e.g., XR software with simultaneous multimodal inputs).

Limitations

After analysis, our project faces the following limitations:

- **Lack of unified evaluation**: Current efforts remain fragmented across subdomains without standardized metrics or comparative baselines.
- **Task collection barrier**: Relevant evaluation tasks are dispersed across disparate research fields, creating significant overhead for researchers.
- **Lack of high-quality data**: Most existing datasets are only annotated and labeled for specific tasks. During the migration process of the dataset, due to the lack of corresponding new annotations (true labels), only a small number of tasks that do not rely on true label evaluation meet the migration conditions.

Future Work From Term 1

- **Refining the taxonomy – Done**
  - Address misclassifications caused by errors or random factors.
  - Ensure no potential research directions are overlooked.
- **Proposing New Tasks – Done**
  - Generalize and expand tasks from existing datasets and task trees.
  - Cover more modalities for broader applicability.
- **Expand Experiment Size – Done**
  - Tested models Size
  - Benchmark Size

Future Work

In the future, we hope to continue improving our work in the following aspects:

- Open-source all materials for follow-up studies.
- Conduct more comprehensive experiments in more tasks on our task taxonomy tree.
- Explore enhancing LMMs' performance by multi-agent tool calling techniques.

Part 7. Q & A

Part 8. References

References

[1] C. Si, Y. Zhang, Z. Yang, R. Liu, and D. Yang, Design2code: How far are we from automating front-end engineering? 2024. arXiv: 2403.03163 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2403.03163.

[2] Z. Liu, C. Chen, J. Wang, Y. Huang, J. Hu, and Q. Wang, "Owl eyes: Spotting ui display issues via visual understanding," in Proceedings of the 35th IEEE/ACM Inter-national Conference on Automated Software Engineering, ser. ASE '20, ACM, Dec. 2020, pp. 398–409. DOI: 10.1145/3324884.3416547. [Online]. Available: http://dx.doi.org/10.1145/3324884.3416547.

[3] J. Chen, M. Xie, Z. Xing, et al., "Object detection for graphical user interface: Old fashioned or deep learning or a combination?" In Proceedings of the 2020 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, New York, NY: ACM, 2020. DOI: 10.1145/3368089.3409691.

References

[4] L. Bao, Z. Xing, X. Xia, D. Lo, M. Wu, and X. Yang, "Psc2code: Denoising code extraction from programming screencasts," ACM Transactions on Software Engineering and Methodology, vol. 29, no. 3, pp. 1–38, Jun. 2020, ISSN: 1557-7392. DOI:10.1145/3392093. [Online]. Available: http://dx.doi.org/10.1145/3392093

[5] S. Li, L. Bu, G. Bai, Z. Guo, K. Chen, and H. Wei, "Vitas: Guided model-based vuitesting of vpa apps," in Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering, ser. ASE '22, Rochester, MI, USA: Association for Computing Machinery, 2023, ISBN: 9781450394758. DOI: 10.1145 /3551349.3556957. [Online]. Available: https://doi.org/10.1145/3551349.3556957.

[6] Z. Sasindran, H. Yelchuri, and T. V. Prabhakar, "Semascore: A new evaluation metric for automatic speech recognition tasks," in Interspeech 2024, ser. interspeech2024, ISCA, Sep. 2024, pp. 4558–4562. DOI: 10.21437/interspeech.2024-2033. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2024-2033.

References

[7] K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, and Z. Wu, "Seeclick: Harnessing gui grounding for advanced visual gui agents," 2024.
[8] K. Li, Z. Meng, H. Lin, Z. Luo, Y. Tian, J. Ma, ... and T.S. Chua, "Screenspot-pro: Gui grounding for professional high-resolution computer use," In Workshop on Reasoning and Planning for Large Language Models, 2025.