Reformulate, Retrieve, Localize: Agents for Repository-Level Bug Localization

Supplementary material

S.1 BM25 Tool-Call Evaluation

How counts were extracted.
Top-k preferences per run/dataset were computed with:

```
# Per-run top-k requests (counts by k)
cat *.jsonl | jq -r 'select(.type == "tool_result" and .bm25_top_k != null) |
.bm25_top_k' | sort | uniq -c
```

S.1.1 Top-k preferences by model/dataset

TOP-1 retrieval (counts of `bm25_top_k` requests)

	SWE: k=10	SWE: k=20	SWE: k=30	LCA: k=10	LCA: k=20	LCA: k=30
Qwen2.5 – Run1 (all)	216	0	76	100	0	46
Qwen2.5 – Run1 (best)	140	160	0	78	0	74
Qwen2.5 – Average	**178**	**80**	**38**	**89**	**0**	**60**
Qwen3 – Run1 (all)	5	234	12	4	112	3
Qwen3 – Run1 (best)	7	184	12	13	172	9
Qwen3 – Average	**6**	**209**	**12**	**8.5**	**142**	**6**

Takeaways (TOP-1):

- **SWE:** Qwen2.5 favors **k=10**; Qwen3 strongly favors **k=20**.
- **LCA:** both models lean toward **k=20**, with Qwen2.5 also using **k=30** frequently for the *best* variation.

TOP-5 retrieval

	SWE: k=10	SWE: k=20	SWE: k=30	LCA: k=10	LCA: k=20	LCA: k=30
Qwen2.5 – Run1 (all)	1	0	299	2	0	171
Qwen2.5 – Run1 (best)	0	0	300	1	0	159
Qwen2.5 – Average	**0.5**	**0**	**299.5**	**1.5**	**0**	**165**
Qwen3 – Run1 (all)	1	252	99	0	149	33
Qwen3 – Run1 (best)	1	117	101	0	128	62
Qwen3 – Average	**1**	**184.5**	**100**	**0**	**138.5**	**47.5**

TOP-10 retrieval

	SWE: k=10	SWE: k=20	SWE: k=30	LCA: k=10	LCA: k=20	LCA: k=30
Qwen2.5 – Run1 (all)	1	0	302	1	0	151
Qwen2.5 – Run1 (best)	0	0	300	1	0	151
Qwen2.5 – Average	**0.5**	**0**	**301**	**1**	**0**	**151**
Qwen3 – Run1 (all)	4	348	24	6	180	5

Qwen3 – Run1 (best)	4	171	47	5	174	21
Qwen3 – Average	**4**	**259.5**	**35.5**	**5.5**	**177**	**13**

Takeaways (TOP-5/10):

- **Qwen2.5** shifts to **k=30** for broader retrieval (TOP-5/10) on **both** datasets.
- **Qwen3** still prefers **k=20** overall, but increases **k=30** usage on SWE for wider recall.

S.1.2 Tool-usage volume and redundancy

How counts were extracted.

```
# Tool usage per type (Qwen2.5 example)
cat qwen2.5*/*.jsonl \
| jq -r 'select(.type == "tool_result")
         | if .bm25_top_k then "bm25_top_k"
           elif .extract_relevant then "extract_relevant"
           elif .view_readme then "view_readme"
           else "unknown" end' \
| sort | uniq -c
```

```
# File views (all vs unique)
cat qwen2.5*/*.jsonl | jq -r 'select(.type == "tool_decision" and .tool_name ==
"view_file") | .tool_args.file_path' | wc -l
cat qwen2.5*/*.jsonl | jq -r 'select(.type == "tool_decision" and .tool_name ==
"view_file") | .tool_args.file_path' | sort -u | wc -l
```

LCA — Run 1

Model	bm25_top_k (Total / Avg per repo)	extract_relevant	view_file	view_readme	Unique file views	Redundancy (view_file ÷ unique)
Qwen2.5-coder 32B	1000 / 6.7	900 / 6.0	5523 / 36.8	0 / 0	800 / 5.3	**6.90×**
Qwen3-coder 30B	1076 / 7.2	900 / 6.0	3326 / 22.2	2 / 0	598 / 4.0	**5.56×**

Observations (LCA):

- Duplication exists but is lower than SWE (≈5.6–6.9×).
- Some repositories with **<10** Python files triggered repeated BM25 calls.

SWE-Bench Lite — Run 1

Model	bm25_top_k (Total / Avg per repo)	extract_ relevant	view_file	view_readme	Unique file views	Redundancy (view_file ÷ unique)
Qwen2.5-coder 32B	1795 / 6.0	1798 / 6.0	12056 / 40.2	0 / 0	599 / 2.0	**20.13×**
Qwen3-coder 30B	1623 / 5.4	1428 / 4.8	6044 / 20.1	1 / 0	465 / 1.6	**13.00×**

Observations (SWE):

- Both models re-open files many times. Qwen2.5 shows **~20×** duplication; Qwen3 **~13×**.
- `view_readme` is effectively unused.

S.1.3 Error conditions

How counts were extracted.

```
# Aborted file view (example)
cat qwen2.5*/*.jsonl | jq -r 'select(.type == "abort file view" and .step == "turn_3")
| .step' | wc -l

# Aborted invalid output (non-path)
cat qwen2.5*/*.jsonl | jq -r 'select(.type == "abort" and .reason == "still invalid
after reprompts") | .reason' | wc -l

# Model timeouts
cat qwen3*/*.jsonl | jq -r 'select(.type == "timeout") | .type' | wc -l
```

LCA — Run 1

Model	Aborted file view	% aborted	Model timeout	Aborted invalid output

Qwen2.5-coder 32B	554	**10.03%**	0	4
Qwen3-coder 30B	127	**3.82%**	11	9

SWE-Bench Lite — Run 1

Model	Aborted file view	% aborted	Model timeout	Aborted invalid output
Qwen2.5-coder 32B	1518	**12.59%**	0	2
Qwen3-coder 30B	384	**6.35%**	18	12

Interpretation:

- **SWE** shows higher abort rates than **LCA**.
- **Timeouts** occur only with **Qwen3**.

S.2 Prompts and Tools

This section records the exact prompts used to steer the agent during bug-localization. Placeholders like `{n}`, `{issue_description}`, `{repo_name}`, `{function_calls}`, `{tool_call}`, `{args}`, `{bm25_retrieved_pretty}`, `{viewed_files_pretty}` are programmatically substituted at runtime.

S.2.1 System prompt

ID	**SYSTEM_PROMPT**
Purpose	Establish expert role and objective; set target number of files {n}.
Template	You are an expert software engineer specialized in bug localization. Your task is to identify the files most likely to contain bugs based on bug descriptions. You have access to several tools to help you in this task. From the files you have seen, you will select {n} files that you believe are most likely to contain the bug.

S.2.2 Turn 1

In this first turn, the agent is given basic information about the repository and a full bug description. It needs to call the ‘extract_relevant’ tool, to move to the next turn.

ID	**PROMPT_TURN_1**
Purpose	Start the task with the issue text and repository name; enumerate tools; require JSON-only output (either a tool call or final ranked list).
Template	<bug_description> {issue_description} </bug_description> The repository name is: <repo_name> {repo_name} </repo_name> You have access to the following tools: {function_calls} You must use these tools to find {n} files that are most likely to contain the bug. Start by analyzing the issue description and then decide which tool to use first. If you are done, provide your final ranked list of {n} files most likely to contain bugs in a JSON array as follows: {"ranked_files": ["file1", "file2", ...]} Otherwise, return only the tool call as JSON. Example: {"function_call": "func_name", "args": {"arg_name": 10} } Reply ONLY with the JSON response.

ID	**FUNCTION_CALLS_TURN_1**
Purpose	Tool options shown in Turn 1.
Template	extract_relevant() -> Retrieve the most relevant info from the bug. Returns a JSON object with explanation, relevant code snippets, error messages and file paths mentioned in the bug.

S.2.3 Turn 2

In this turn, the agent is prompted to call the BM25 tool with k={10,20,30}.

ID	**PROMPT_TURN_2**
Purpose	Choose BM25 breadth or finalize; JSON-only (tool call or final list).

Template	Based on the tool call results, please select your next action. Recall that you need to provide the {n} files most likely to contain bugs. <tool_called> {tool_call} </tool_called> <args> {args} </args> You have access to the following tools: {function_calls} If you are done, provide your final ranked list of {n} files most likely to contain bugs in a JSON array as follows: {"ranked_files": ["file1", "file2", ...]} Otherwise, return only the tool call as JSON. Example: {"function_call": "func_name", "args": {"arg_name": 10} } Reply ONLY with the JSON response.

ID	**FUNCTION_CALLS_TURN_2**
Purpose	BM25 retrieval options.
Template	get_bm25_top10() -> Retrieve the top-10 best results with BM25 search. get_bm25_top20() -> Retrieve the top-20 best results with BM25 search. get_bm25_top30() -> Retrieve the top-30 best results with BM25 search.

S.2.4 Turn 3

This turn allows the agent to iteratively view chunks of files it selects. It has to provide a valid file path to view. The prompt “`SELF_EVAL_POST_VIEW_PROMPT`” was added while the agent is viewing files so that it does not attempt to complete the code in the viewed files. As it’s only viewing partial content, earlier versions attempted to complete the code endlessly.

ID	**PROMPT_TURN_3**
Purpose	Inspect files or finalize; JSON-only; caution against completing code.

Template	Based on the tool call results, please select your next action. Recall that you need to provide the {n} files most likely to contain bugs. <tool_called> {tool_call} </tool_called> <args> {args} </args> You have access to the following tools: {function_calls} If you are done, provide your final ranked list of {n} files most likely to contain bugs in a JSON array as follows: {"ranked_files": ["file1", "file2", ...]} Otherwise, return only the tool call as JSON. Example: {"function_call": "func_name", "args": {"arg_name": 10} } Reply ONLY with the JSON response. Remember, do not try to complete the code.

ID	**FUNCTION_CALLS_TURN_3**
Purpose	Viewing tools used during Turn 3.
Template	view_file(`file_path`) -> View the content of the specified file. view_readme() -> View the content of the README file in the repository.

ID	**SELF_EVAL_POST_VIEW_PROMPT**
Purpose	Post-view file prompt. Used to instruct the model to not try to complete the file after the tool call has returned a chunk of the file, and show the options available next: return a ranked list of files or view another file.
Template	You are analyzing code. The code may be truncated. DO NOT attempt to complete incomplete functions or classes. If you see incomplete code, work with what's available. Select your next action. Remember you need to provide the {n} files most likely to contain bugs. You may reply with ONE of these JSON shapes ONLY: A) {{"ranked_files": ["file1", "file2", ...], "confidence_scores": ["score1", "score2", ...]}} B) {{"function_call":"view_file","args":{{"file_path":string}}}} Never include code blocks, file contents, or extra keys.

	Reply ONLY with the required JSON format.

S.2.5 Error-handling prompts

Those prompts are issued whenever the agent provides an invalid path to view a file, repeated views of the same file, or provides a ranked files list with invalid files.

ID	**PROMPT_TOOL_ERROR_FILE_PATH**
Purpose	Recover from invalid view_file path by requesting a valid call; JSON-only.
Template	The last tool call failed. <tool_called> {tool_call} </tool_called> <args> {args} </args> <error> {error} </error> Call the tool view_file again with a valid path. Reminder of tool documentation: view_file(\`file_path\`) -> View the content of the specified file. Reply ONLY with the JSON response.

ID	**PROMPT_RANKED_FILES_INVALID**
Purpose	Repair final answer when some paths are invalid; JSON-only corrected list.

Template	Some of the file paths in your last ranked_files are invalid for this repository. <invalid_paths> {invalids_pretty} </invalid_paths> Please return ONLY a corrected JSON object containing {n} files ranked from most to least likely to contain bugs, in the format: {"ranked_files": ["path1", "path2", ...]}

ID	**PROMPT_MAX_FILE_VIEWS**
Purpose	Stop excessive viewing of the same file and force finalization; JSON-only final list.
Template	You have reached the maximum number of allowed file views. Use the information retrieved previously to construct your final ranked list of {n} files most likely to contain bugs in a JSON array as follows: {"ranked_files": ["file1", "file2", ...]} Reply ONLY with the JSON response.

S.2.6 Answer review (post-hoc scoring)

This is the last stage of the agent conversation, where it has provided a list of ranked files. This set of prompts reconstructs the context from scratch and asks the agent to evaluate its answer and re-rank the files it provided.

ID	**ANSWER_REVIEW_SYSTEM_PROMPT**
Purpose	Set role and scope for reviewing a proposed ranked list with available evidence.
Template	You are an expert software engineer specialized in bug localization. Your task is to review a ranked list of files most likely to contain bugs based on a bug description. You have access to the evidence seen during the investigation, such as file views and BM25 retrievals. Your goal is to review and/or adjust the ranked list of files based on your confidence that each file contains the bug.

ID	**ANSWER_REVIEW_PROMPT_W_VIEW**
Purpose	Score and possibly re-rank final list; output must include ranked_files and confidence_scores.
Template	This is the bug description: <bug_description> {issue_description} </bug_description> BM25 retrieved the following files: <bm25_retrieved> {bm25_retrieved_pretty} </bm25_retrieved> You have also viewed the following files during your investigation: <viewed_files> {viewed_files_pretty} </viewed_files> \n This is the final ranked list of files you provided: {answer}. You are required to return {n} files most likely to contain bugs. Provide a confidence score for each file in the list based on this rubric: <RUBRIC> 9-10 (Near-certain): Direct file mention in bug description or stack frame, you have seen the file content and it matches the bug context. 7-8 (Strong): Clear functional ownership and multiple strong signals (identifiers + behavior + BM25 retrieval). 5-6 (Plausible): Good topical/structural match with partial identifier/behavior evidence. 3-4 (Weak): Mostly retrieval/path proximity with little concrete alignment, no evidence. 1-2 (Very unlikely): Invalid/irrelevant, or only superficial similarity, no evidence. </RUBRIC> Adjust the ranking based on your confidence scores. If your confidence score is below 7 for the first 2-3 files, you should consider replacing it with a file for which you have higher confidence. You may reply with ONE of these JSON shapes ONLY:

	A) {{"ranked_files": ["file1", "file2", ...], "confidence_scores": ["score1", "score2", ...]}} B) {{"function_call":"view_file","args":{{"file_path":string}}}} Never include code blocks, file contents, or extra keys. If you are about to paste file content, STOP and return JSON instead. Reply ONLY with the required JSON format.

S.2.8 Expected Output Summary

Tool-call output (from the model):

```
{"function_call": "func_name", "args": {"arg_name": 10}}
```

Final answer (ranked files):

```
{"ranked_files": ["path1", "path2", "..."]}
```

Reviewed answer with confidence:

```
{"ranked_files": ["path1", "path2", "..."], "confidence_scores": ["9","7","6", "..."]}
```

Validation rules: reply must be *only* JSON. File paths must exist in the target repository; invalid paths trigger `PROMPT_RANKED_FILES_INVALID`. Excessive `view_file` calls trigger `PROMPT_MAX_FILE_VIEWS`.