M1 Internship Defense

DynPol : Dynamic Policy Library for web agents

Martin CUINGNET

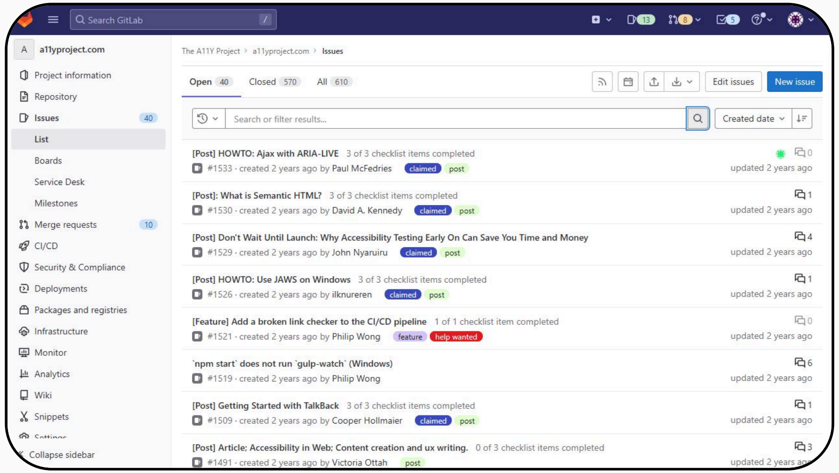
Aldo LIPANI, Jerome RAMOS, Bin WU | Web Intelligence Team, UCL

2025-12-05

ENS Paris-Saclay

1. Context

1.1 What is a web agent ?



[Post] HOWTO: Ajax with ARIA-LIVE 3 of 3 checklist items completed 0
#1533 · created 2 years ago by Paul McFedries claimed post updated 2 years ago

[Post]: What is Semantic HTML? 3 of 3 checklist items completed 1
#1530 · created 2 years ago by David A. Kennedy claimed post updated 2 years ago

[Post] Don't Wait Until Launch: Why Accessibility Testing Early On Can Save You Time and Money 4
#1529 · created 2 years ago by John Nyaruiru claimed post updated 2 years ago

[Post] HOWTO: Use JAWS on Windows 3 of 3 checklist items completed 1
#1526 · created 2 years ago by ilknureren claimed post updated 2 years ago

[Feature] Add a broken link checker to the CI/CD pipeline 1 of 1 checklist item completed 0
#1521 · created 2 years ago by Philip Wong feature help wanted updated 2 years ago

`npm start` does not run `gulp-watch` (Windows) 6
#1519 · created 2 years ago by Philip Wong updated 2 years ago

[Post] Getting Started with TalkBack 3 of 3 checklist items completed 1
#1509 · created 2 years ago by Cooper Hollmaier claimed post updated 2 years ago

[Post] Article; Accessibility in Web; Content creation and ux writing. 0 of 3 checklist items completed 3
#1491 · created 2 years ago by Victoria Ottah post updated 2 years ago

Figure 1: Web Environment

⇒ Web Agent ⇒

Action

- `click [380]`
- `type [67] [Carnegie Mellon]`

1.2 Web Navigation Example

Example of web task :

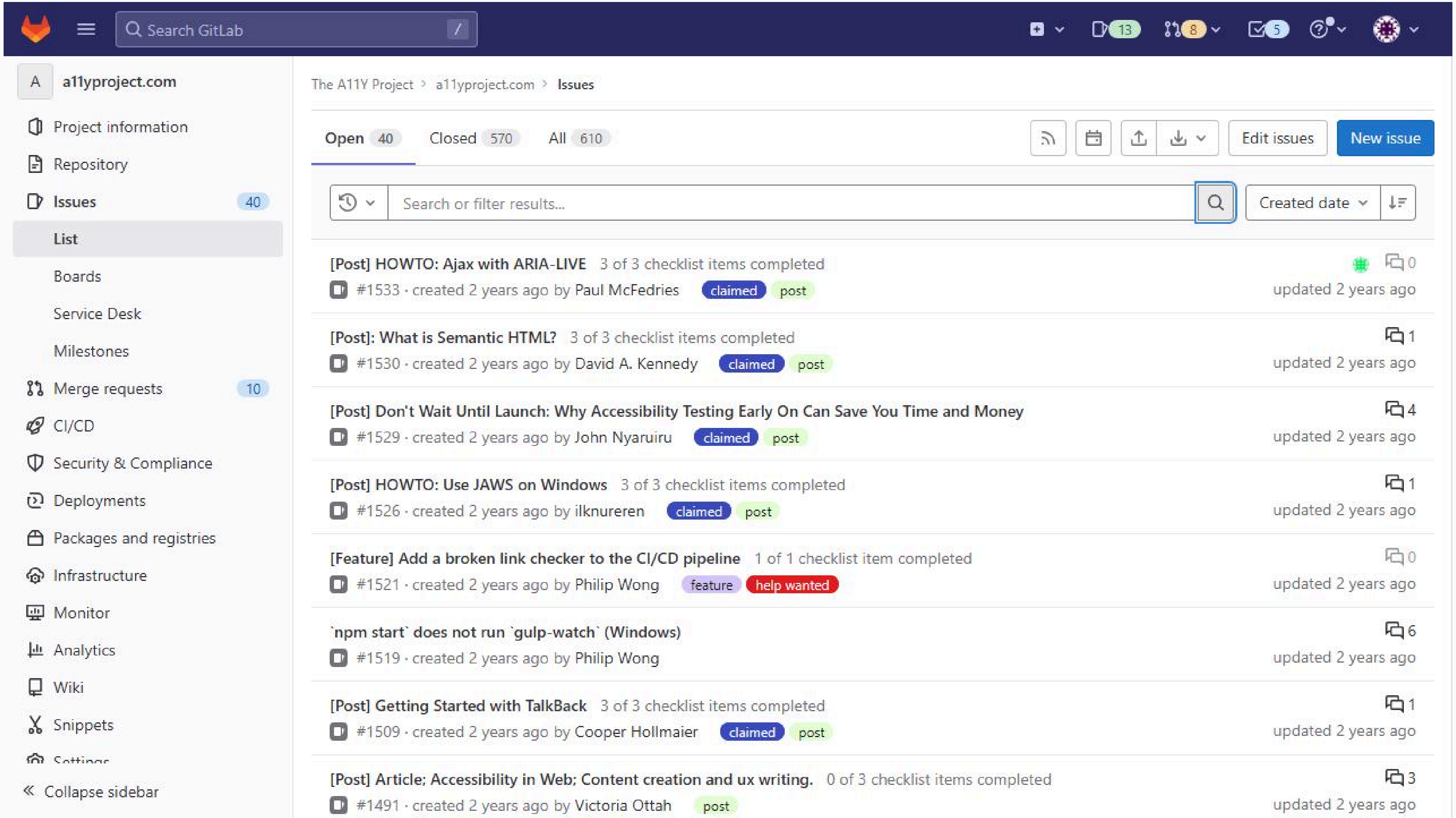
Find the highest rated book on the Shopping site

Sort issues based on a label in a GitLab repo

The agent needs to:

- Find the associated repo
- Go to the issues page
- Sort issues based on the label

1.2 Web Navigation Example



Search GitLab /

13 8 5

A a11yproject.com

- Project information
- Repository
- Issues 40
  - List
  - Boards
  - Service Desk
  - Milestones
- Merge requests 10
- CI/CD
- Security & Compliance
- Deployments
- Packages and registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- Snippets
- Settings

« Collapse sidebar

The A11Y Project › a11yproject.com › Issues

Open 40 Closed 570 All 610

Edit issues New issue

Search or filter results... Created date

[Post] HOWTO: Ajax with ARIA-LIVE 3 of 3 checklist items completed
#1533 · created 2 years ago by Paul McFedries claimed post 0
updated 2 years ago

[Post]: What is Semantic HTML? 3 of 3 checklist items completed
#1530 · created 2 years ago by David A. Kennedy claimed post 1
updated 2 years ago

[Post] Don't Wait Until Launch: Why Accessibility Testing Early On Can Save You Time and Money
#1529 · created 2 years ago by John Nyaruiru claimed post 4
updated 2 years ago

[Post] HOWTO: Use JAWS on Windows 3 of 3 checklist items completed
#1526 · created 2 years ago by ilknureren claimed post 1
updated 2 years ago

[Feature] Add a broken link checker to the CI/CD pipeline 1 of 1 checklist item completed
#1521 · created 2 years ago by Philip Wong feature help wanted 0
updated 2 years ago

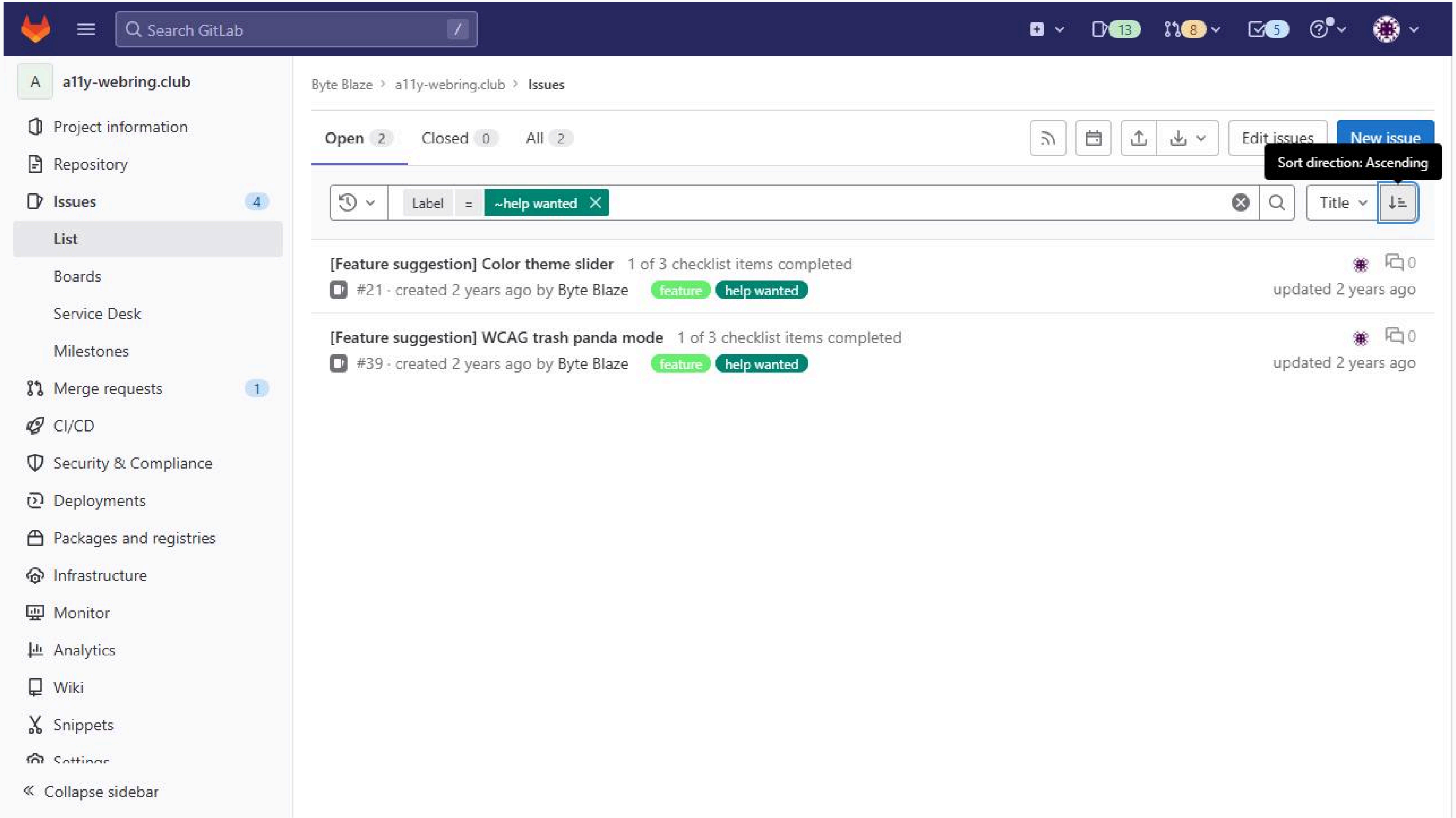
`npm start` does not run `gulp-watch` (Windows)
#1519 · created 2 years ago by Philip Wong 6
updated 2 years ago

[Post] Getting Started with TalkBack 3 of 3 checklist items completed
#1509 · created 2 years ago by Cooper Hollmaier claimed post 1
updated 2 years ago

[Post] Article; Accessibility in Web; Content creation and ux writing. 0 of 3 checklist items completed
#1491 · created 2 years ago by Victoria Ottah post 3
updated 2 years ago

Figure 2: Issue page before sorting

1.2 Web Navigation Example



[Feature suggestion] Color theme slider 1 of 3 checklist items completed
#21 · created 2 years ago by Byte Blaze feature help wanted 0 updated 2 years ago

[Feature suggestion] WCAG trash panda mode 1 of 3 checklist items completed
#39 · created 2 years ago by Byte Blaze feature help wanted 0 updated 2 years ago

Figure 3: Issue page after successfully sorting by clicking on the `help wanted` tag

1.3 Task Examples

Task ID	Website	Task Objective
181	Gitlab	Open my latest created issue that has theme editor in its title to check if it is closed
298	Shopping Admin	Show the most recent completed order
386	Shopping	What is the rating of Ugreen lightning to 3.5mm cable. Please round to the nearest whole number
643	Reddit	Post a notice on a virtual meetup for racing cars enthusiasts on Oct 21st in the nyc subreddit
767	Maps	Find the walkway to the closest chain grocessory owned by a local business from 401 Shady Ave, Pittsburgh.

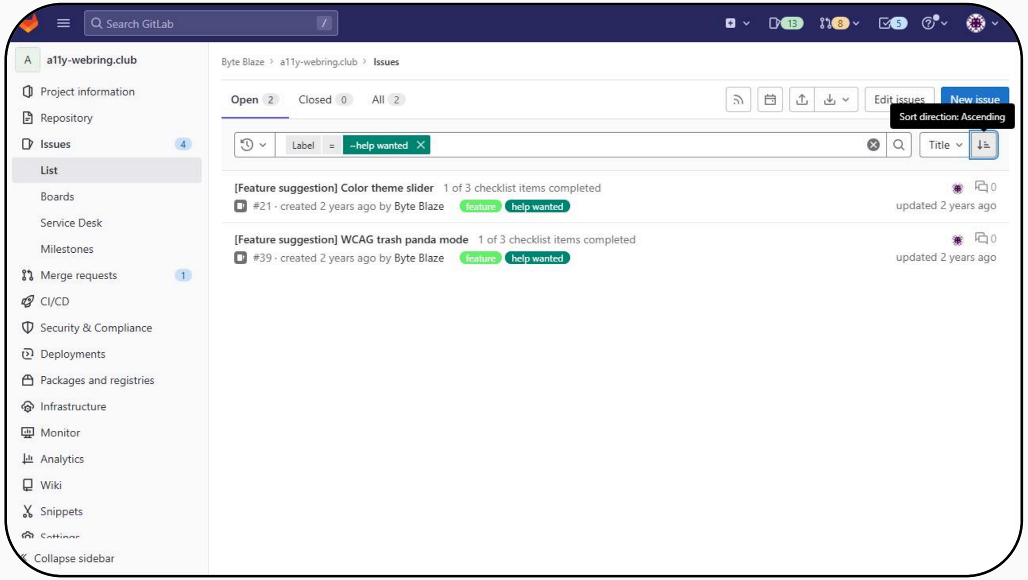
1.4 Challenges with tasks in web environments

Ambiguity with web structures

- Complex objects
- Variety of websites
- Action effect unclear

Complex tasks composed of multiple subtasks

- Complex tasks can be divided in multiple, simpler subtasks
- Hard to keep track of what to do without being lost



Label = ~help wanted

[Feature suggestion] Color theme slider 1 of 3 checklist items completed
#21 · created 2 years ago by Byte Blaze feature help wanted
updated 2 years ago

[Feature suggestion] WCAG trash panda mode 1 of 3 checklist items completed
#39 · created 2 years ago by Byte Blaze feature help wanted
updated 2 years ago

- Find the associated repo
- Go to the issues page
- Sort issues based on the label

1.5 How to deal with these challenges : the SteP solution

SteP (Stacked LLM Policies for Web Actions) [1] : a web-agent paper

Proposed solutions :

- Complex tasks : Simpler subtasks
- Ambiguity : External knowledge

Combine the two solutions ⇒ **policies**

1.6 What is a policy

Policy : a **strategy** to solve a **specific subtask**.

- Contains **instructions** to solve the subtask
- **Called** by the agent and handled **independently**
- Works like a **non-deterministic subroutine**

Example of policy :

`filter_and_sort_issues`

```
Click the button to reveal the sorting options.\n*   Select the desired sorting
criteria from the dropdown menu.\n*   Check the current sort direction. If the
current sort direction is not the desired one, click the sort direction button to
toggle it.\nPlease, use ONLY page operation and no subroutine actions.",
```

In the previous example the policy can be called with : `filter_and_sort_issues [help wanted]`

1.7 SteP policies

SteP policies: **handcrafted in advance**

Problems :

- **Static** Policy Library : No adaptability
- **Handcrafted** policies : Misalignment

1.8 New Components

Solution : Make the policy library **dynamic**

Two components to do so :

- **Self-Improvement Mechanism** : Experience ⇒ Policy instruction optimization
- **Automatic Curriculum** : Planning + Create new policies

2. What is DynPol ?

2.1 Policy Stack

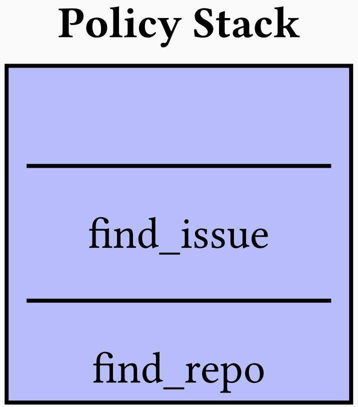


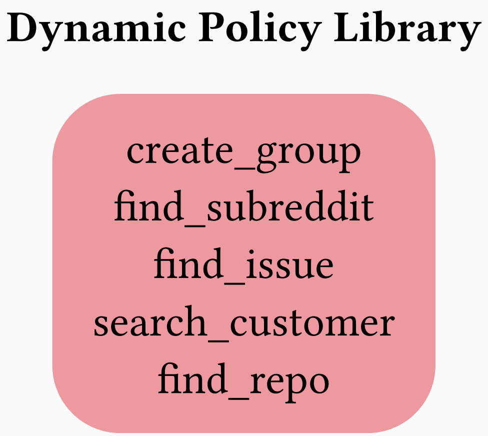
Figure 4: Policy Stack

Policy **currently active** = Policy on **top of the stack**

Calling a policy ⇒ adding policy on top of the stack

Finishing a policy ⇒ popping top policy of the stack

2.2 Policy Library



Dynamic Policy Library

create_group
find_subreddit
find_issue
search_customer
find_repo

Figure 5: Policy Library

Store all **available** policies.

Two ways to edit it:

- **Self-Improvement Mechanism** ⇒ Improve policy instruction
- **Automatic Curriculum** ⇒ Create new policy

2.3 Relevant Policies

Relevant Policies

Figure 6: Relevant Policies

$|\text{LLM Context Size}| < \infty \Rightarrow$ Fixed number of policies passed to agent

How to decide which policies to fetch ?

$\mapsto$ Policy embedding closest to Task embedding

2.4 Main Loop

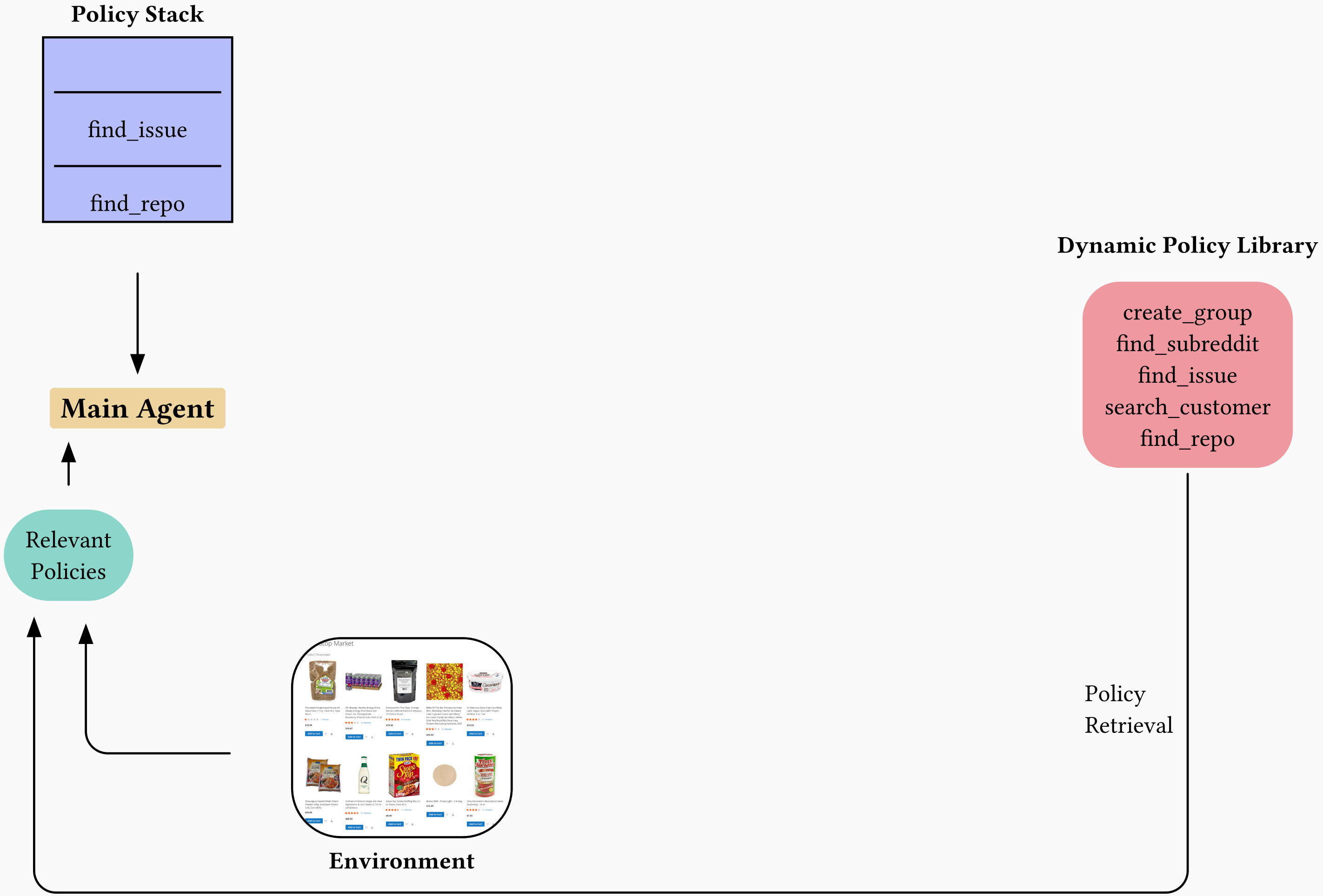
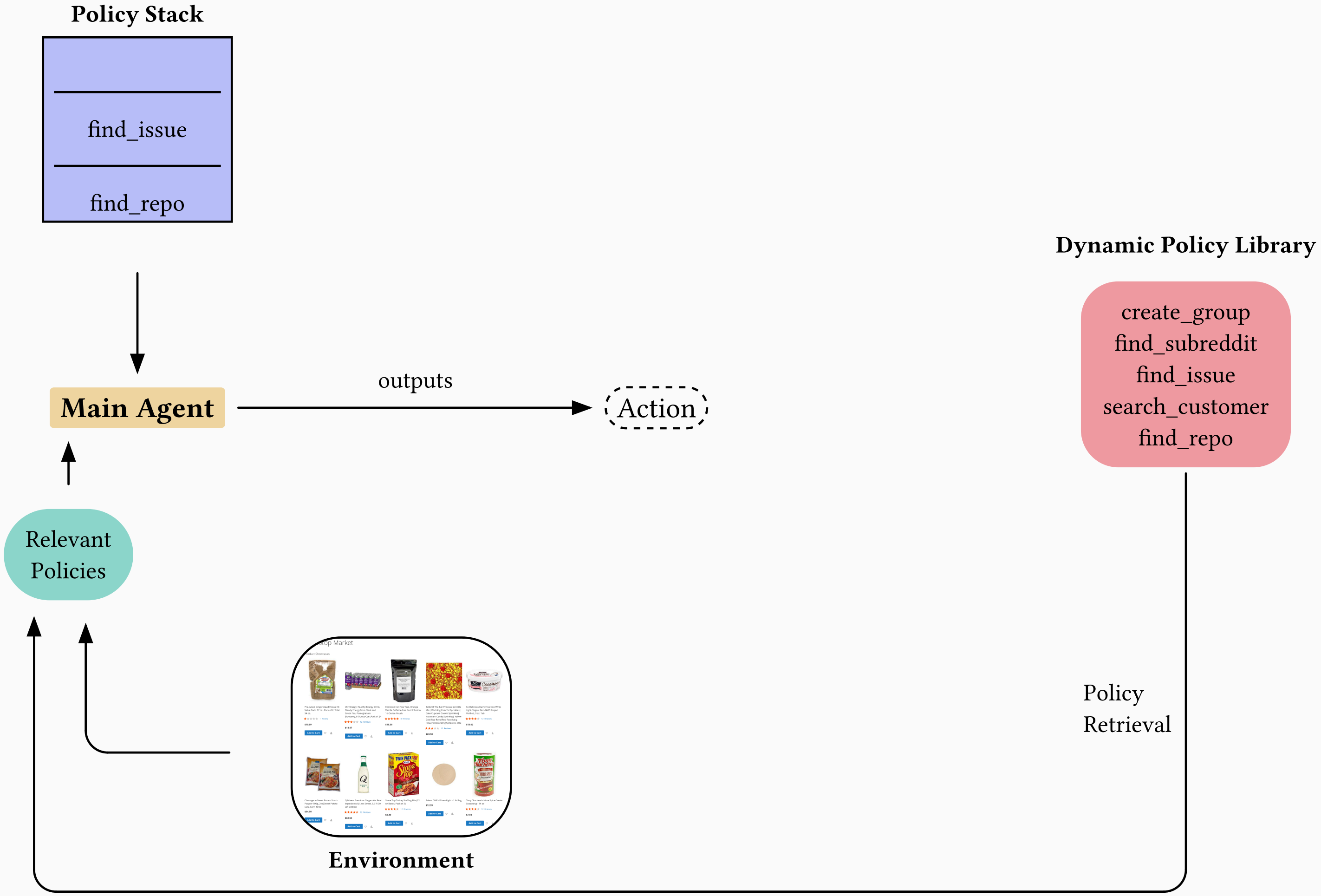


Figure 7: DynPol Main Loop - Step 1

Passed to the **Main Agent LLM** :

- Currently active policy
- Relevant policies
- Environment (html code, task objective,...)

2.4 Main Loop



Policy Stack

find_issue

find_repo

Main Agent

outputs

Action

Relevant Policies

Environment

Dynamic Policy Library

create_group
find_subreddit
find_issue
search_customer
find_repo

Policy Retrieval

Main Agent issues actions to solve the **task objective**

Figure 8: DynPol Main Loop - Step 2

2.4 Main Loop

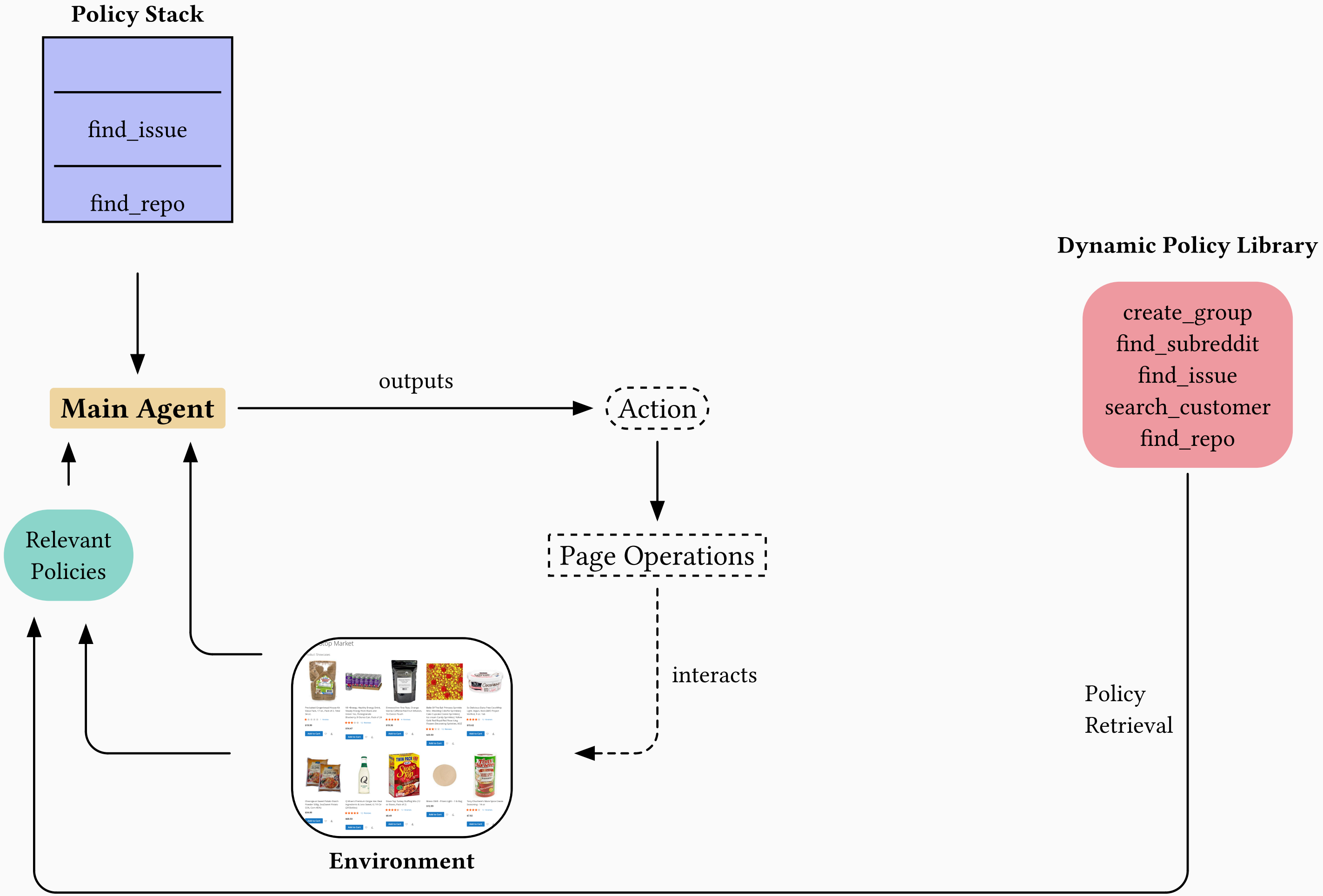


Figure 9: DynPol Main Loop - Step 3

First type of action :
Page Operation

Atomic action that changes (or not) the environment.
Examples :

- `click [380]`
- `type [67] [Carnegie Mellon]`

2.4 Main Loop

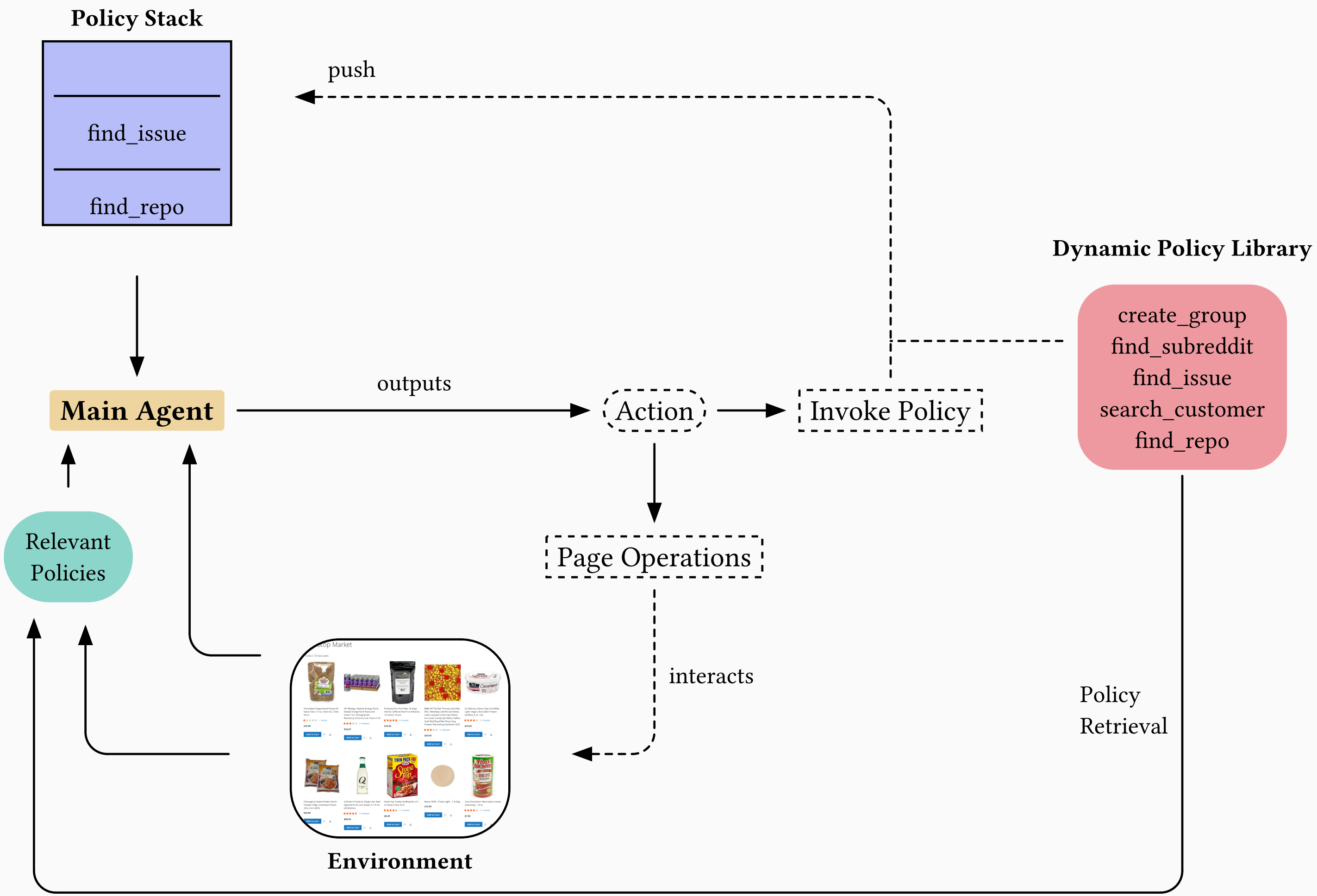


Figure 10: DynPol Main Loop - Step 4

Second type of action : **Invoke a policy**

- Call a policy from **Relevant Policies**
- Add it on top of the stack

2.4 Main Loop

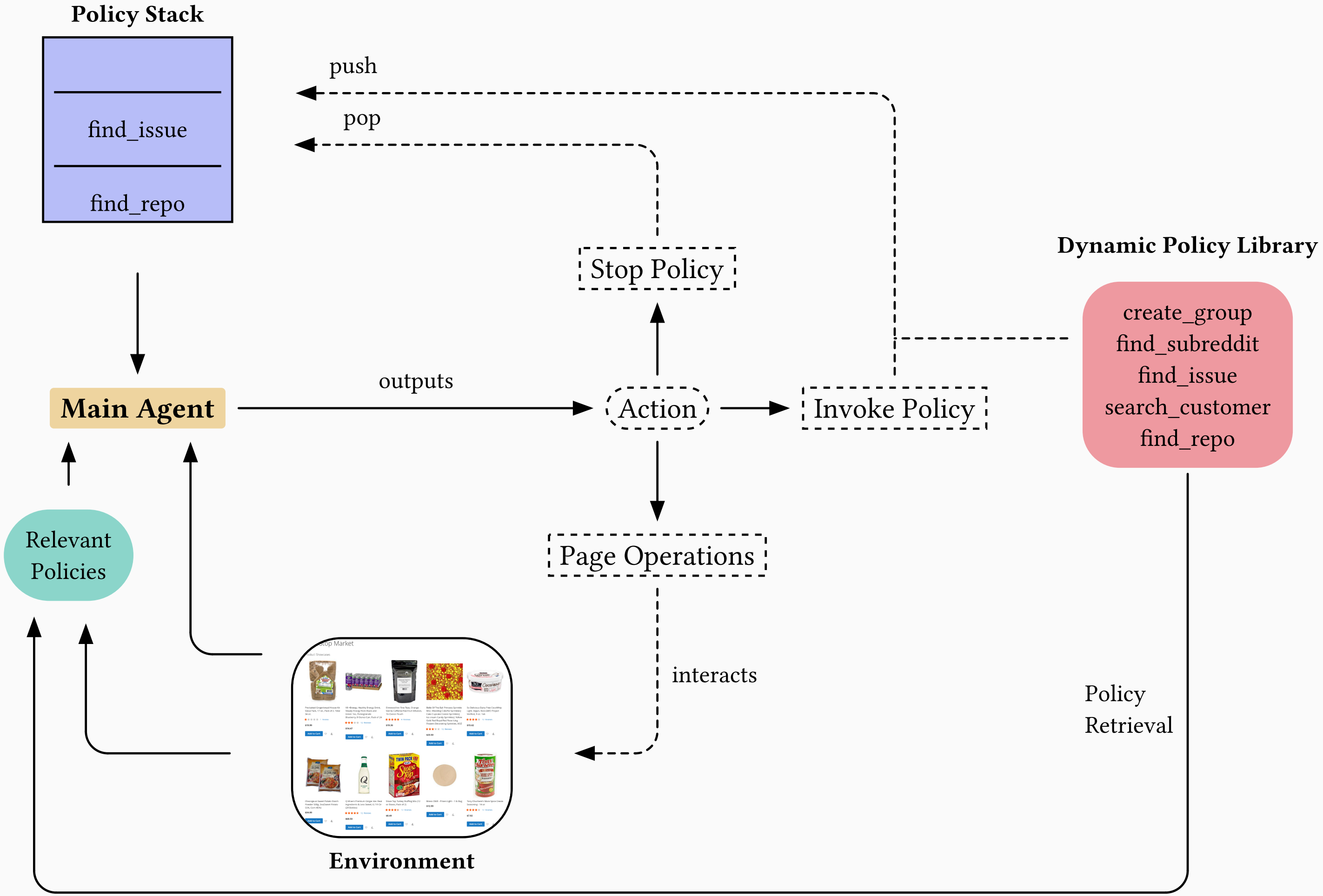
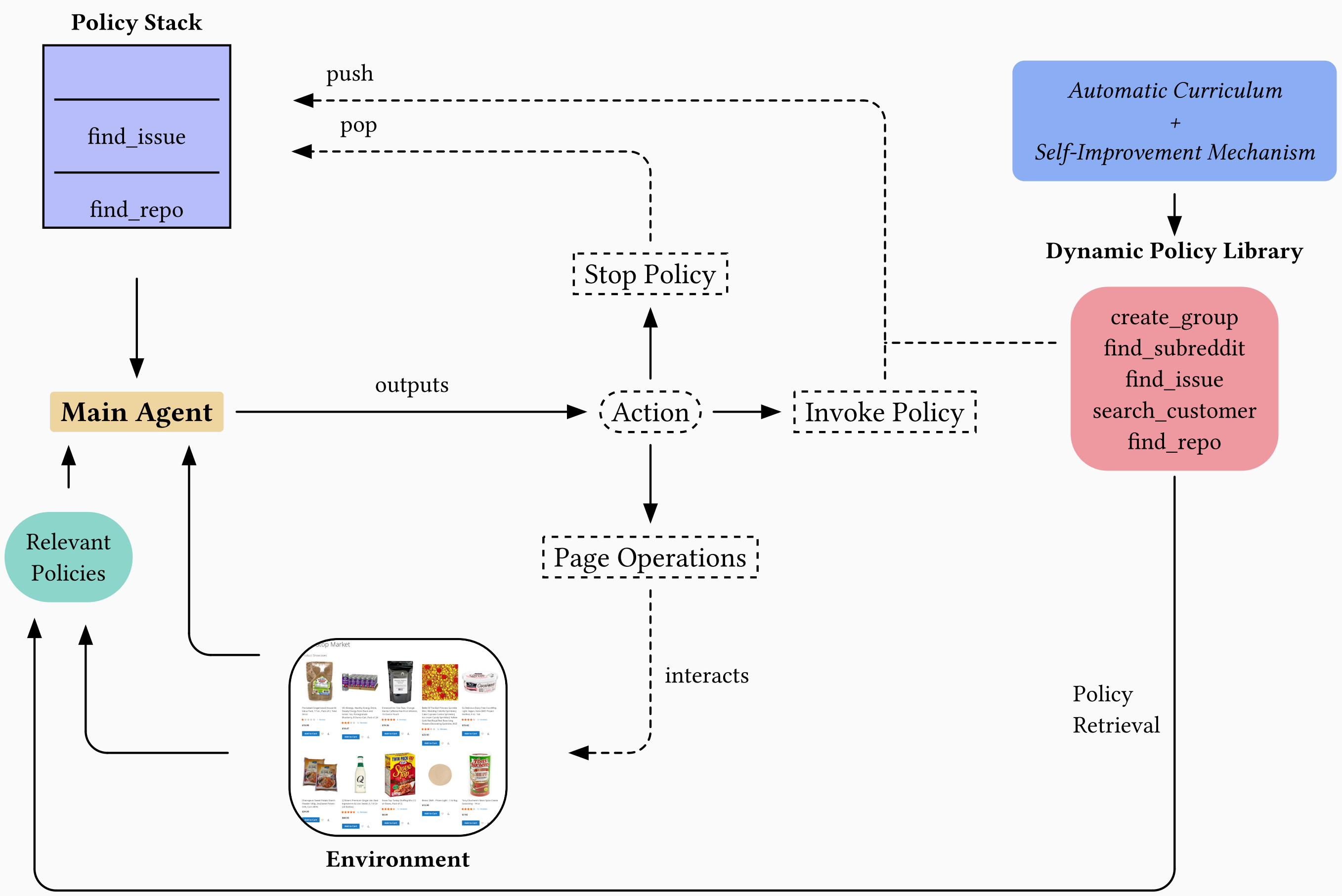


Figure 11: DynPol Main Loop - Step 5

Third type of action : **Stop the currently active policy**

- **Pop** policy off the stack
- Return an answer
- If root policy : **terminates** interaction

2.4 Main Loop



Policy Stack

find_issue

find_repo

push

pop

Automatic Curriculum
+
Self-Improvement Mechanism

Dynamic Policy Library

Stop Policy

create_group
find_subreddit
find_issue
search_customer
find_repo

Main Agent

outputs

Action

Invoke Policy

Relevant Policies

Page Operations

interacts

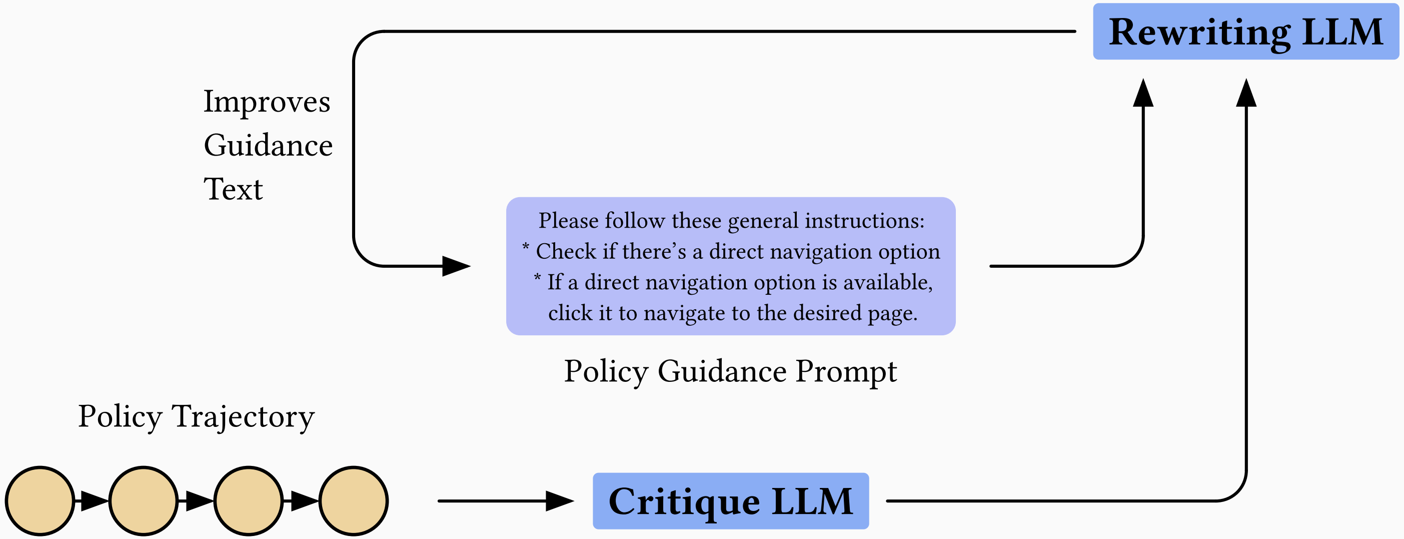
Policy Retrieval

Environment

Figure 12: DynPol Main Loop - Step 5

- After finishing a task : **Self-Improvement Mechanism** : correct polices based on **interaction**
- Before starting a task : **Automatic Curriculum** : plan interaction and create new policies based on **environment**

2.5 Self-Improvement Mechanism



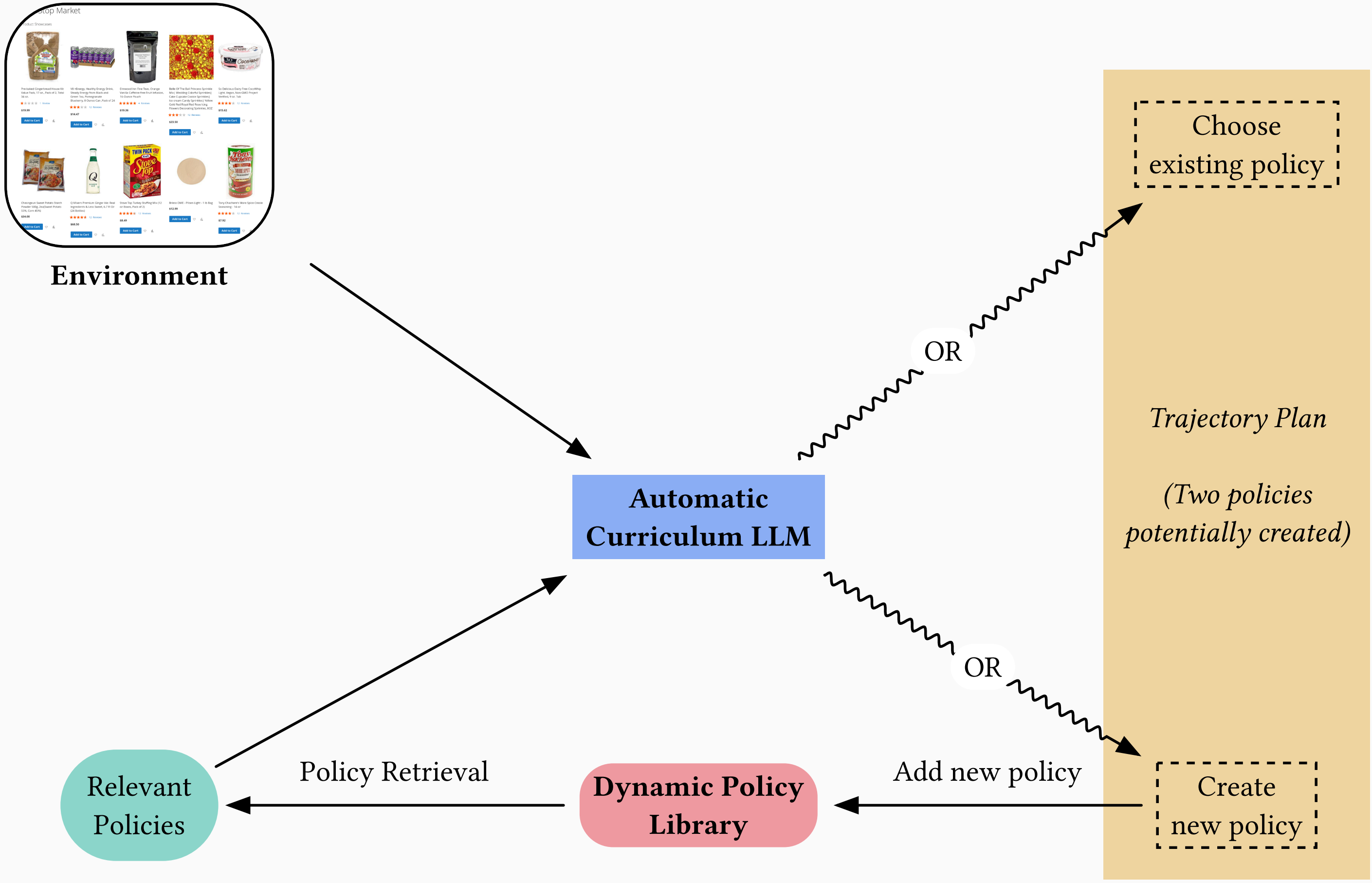
Please follow these general instructions:
* Check if there’s a direct navigation option
* If a direct navigation option is available, click it to navigate to the desired page.

Figure 13: Self-Improvement Mechanism

Trajectory + **Critique LLM** ⇒ Trajectory Breakdown

Trajectory Breakdown + Previous Instruction + **Rewriting LLM** ⇒ New instructions

2.6 Automatic Curriculum



Environment

Automatic Curriculum LLM

OR

OR

Choose existing policy

Trajectory Plan

(Two policies potentially created)

Create new policy

Add new policy

Dynamic Policy Library

Policy Retrieval

Relevant Policies

Figure 14: Automatic Curriculum

Environment + Relevant Policies + **Automatic Curriculum LLM** ⇒ Plan

Plan ⇒ Policies to solve it (either new or in **Relevant Policies**)

2.7 Workflow

Improve DynPol through **experience**

More tasks encountered ⇒ More policies in the **library** ⇒ Better performance

Run DynPol on as many tasks as possible

Run DynPol on the same task multiple times:

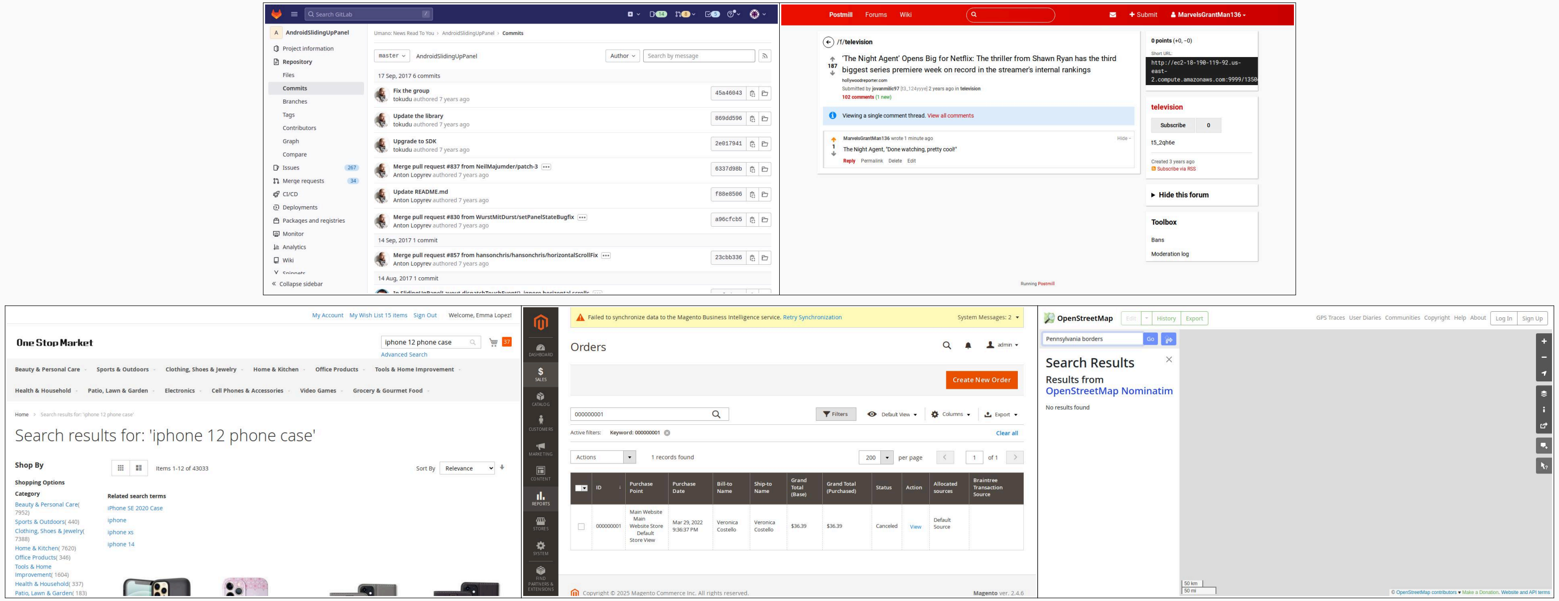
- Better Refinement
- Track policies progress

3. Technical Details

3.1 Testing Environments

Webarena

- 5 types of websites, 812 tasks along with solutions
- Self-Hosted, Closed, Controlled Environment (using AWS)



AndroidSlidingUpPanel
Commits
Postmill
Forums
Wiki
/f/television
'The Night Agent' Opens Big for Netflix: The thriller from Shawn Ryan has the third biggest series premiere week on record in the streamer's internal rankings
One Stop Market
Search results for: 'iphone 12 phone case'
Orders
Create New Order
OpenStreetMap
Search Results
Results from OpenStreetMap Nominatim
No results found

Figure 15: From Left to Right, Top to Bottom :

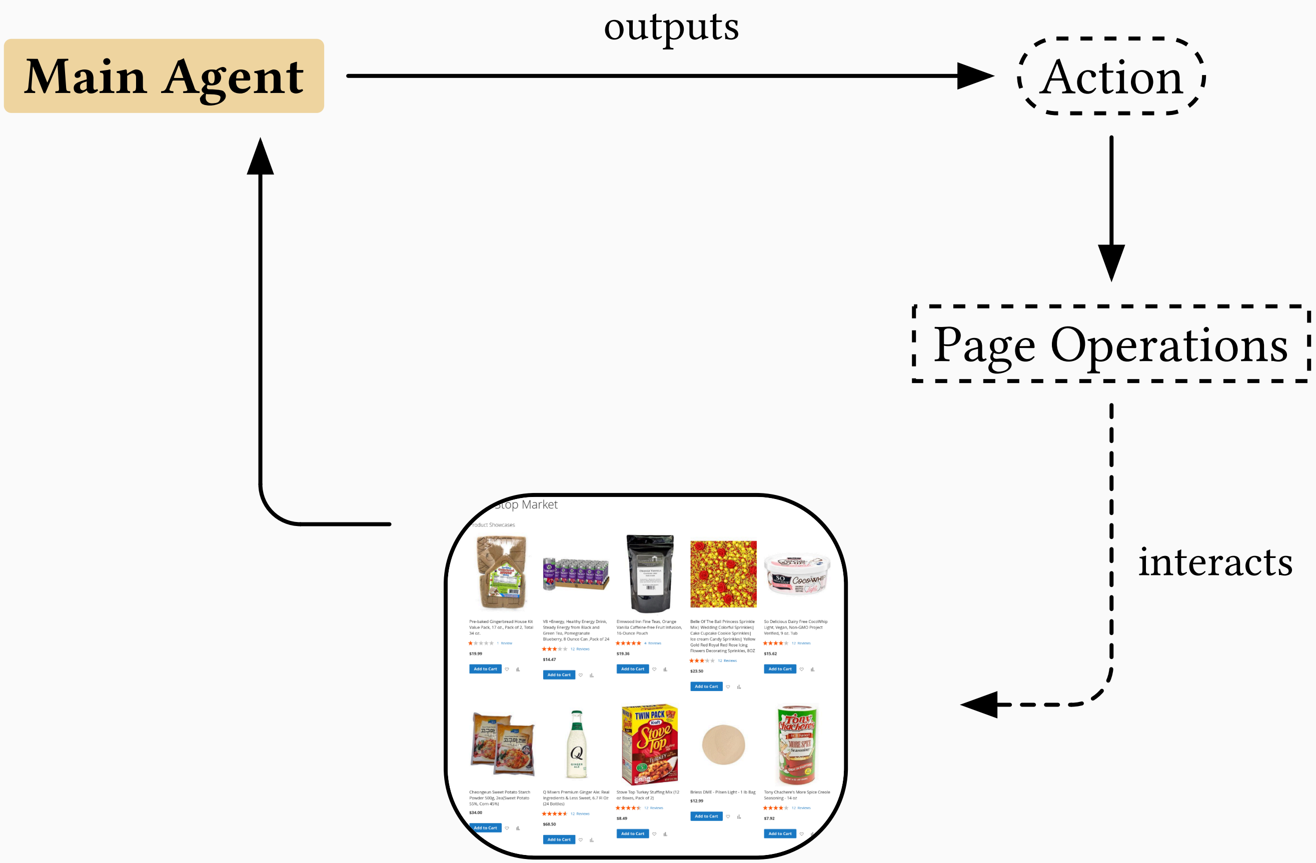
GitLab, Reddit, Shopping, Shopping Admin, Maps (not working)

3.2 Tools used

- **Main LLM** for experiments : `Llmama 3.3 Instruct 70B`
- LLM ran on the Marenostrum BSC with slurm
- Implementation made in python using the **Gymnasium** and **BrowserGym** libraries to interact with web agents

4. Baseline Comparison

4.1 Single LLM



Main Agent

outputs

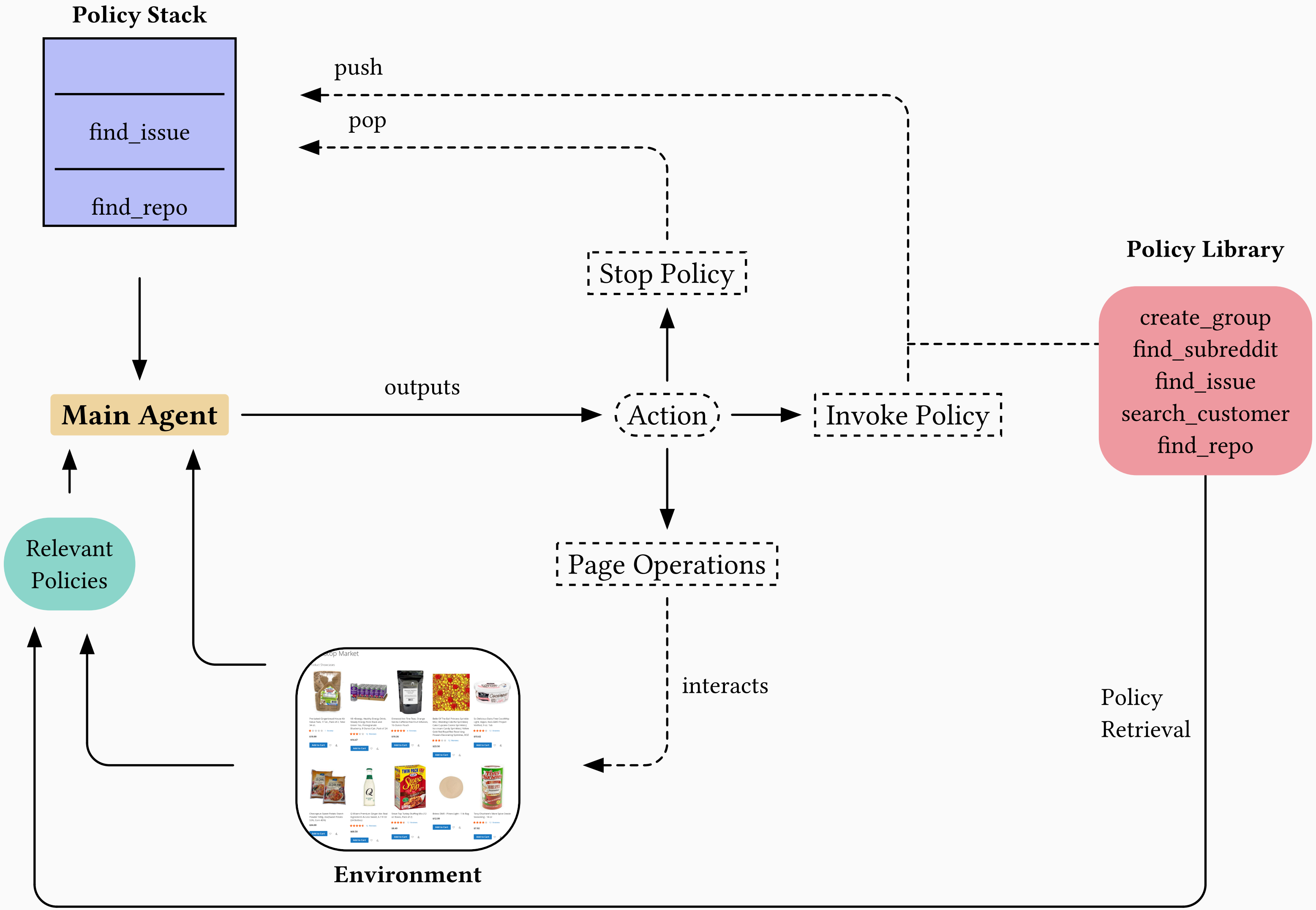
Action

Page Operations

interacts

Environment

4.2 SteP


Policy Stack
find_issue
find_repo
push
pop
Stop Policy
Policy Library
create_group
find_subreddit
find_issue
search_customer
find_repo
Main Agent
outputs
Action
Invoke Policy
Relevant Policies
Page Operations
interacts
Policy Retrieval
Environment

4.3 DynPol and Variants

All variants of DynPol are initialised with the **SteP policies**

Architectures used as baselines :

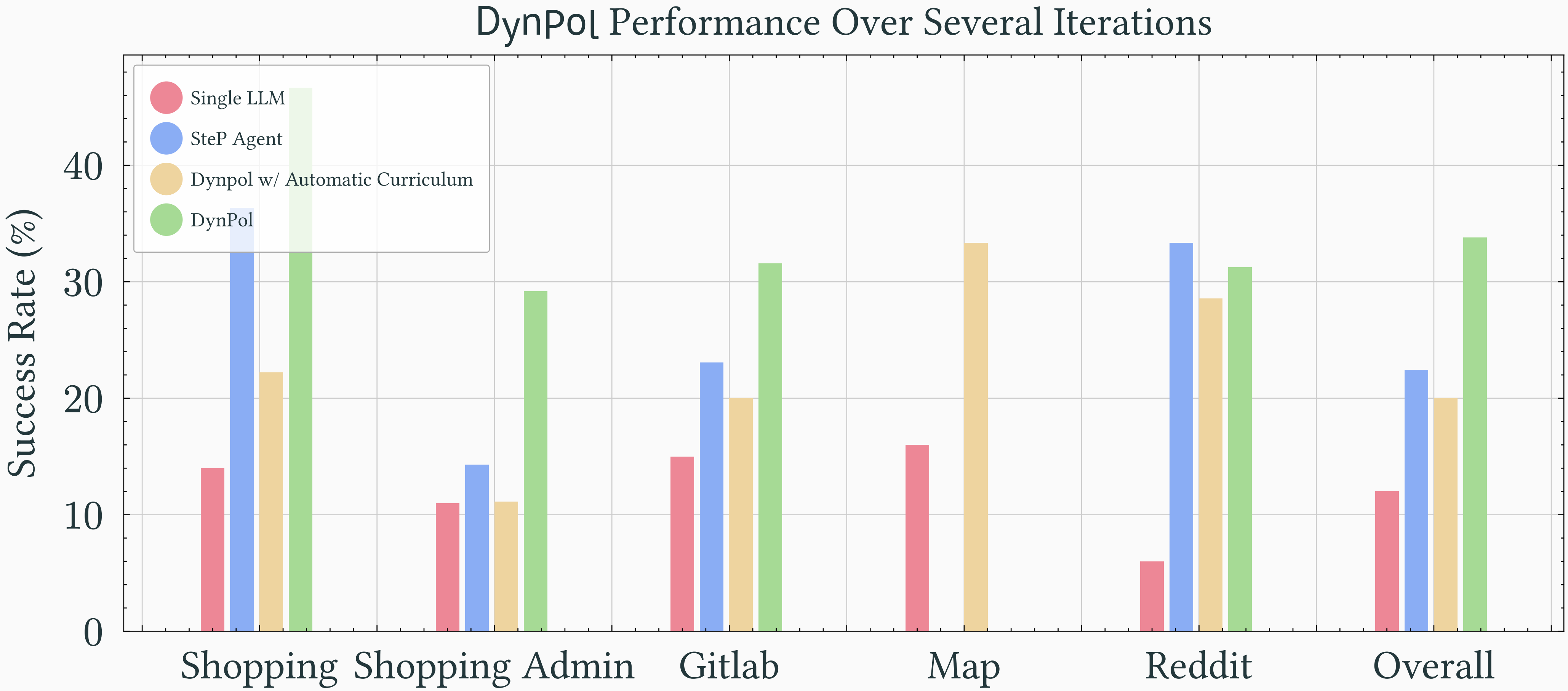
- DynPol without the automatic curriculum (i.e. only optimizing the SteP policies)
- The DynPol architecture as presented before

4.4 Results

Tasks	**Single LLM**	**SteP Agent**	**DynPol without Automatic Curriculum (3 iterations)**	**DynPol (3 iterations)**
Shopping	0.14	0.36	0.22	**0.47**
Shopping Admin	0.11	0.14	0.11	**0.29**
Gitlab	0.15	0.23	0.2	**0.32**
Map	0.16	0	**0.33**	0
Reddit	0.06	0.33	0.29	**0.31**
Overall	0.12	0.22	0.2	**0.34**

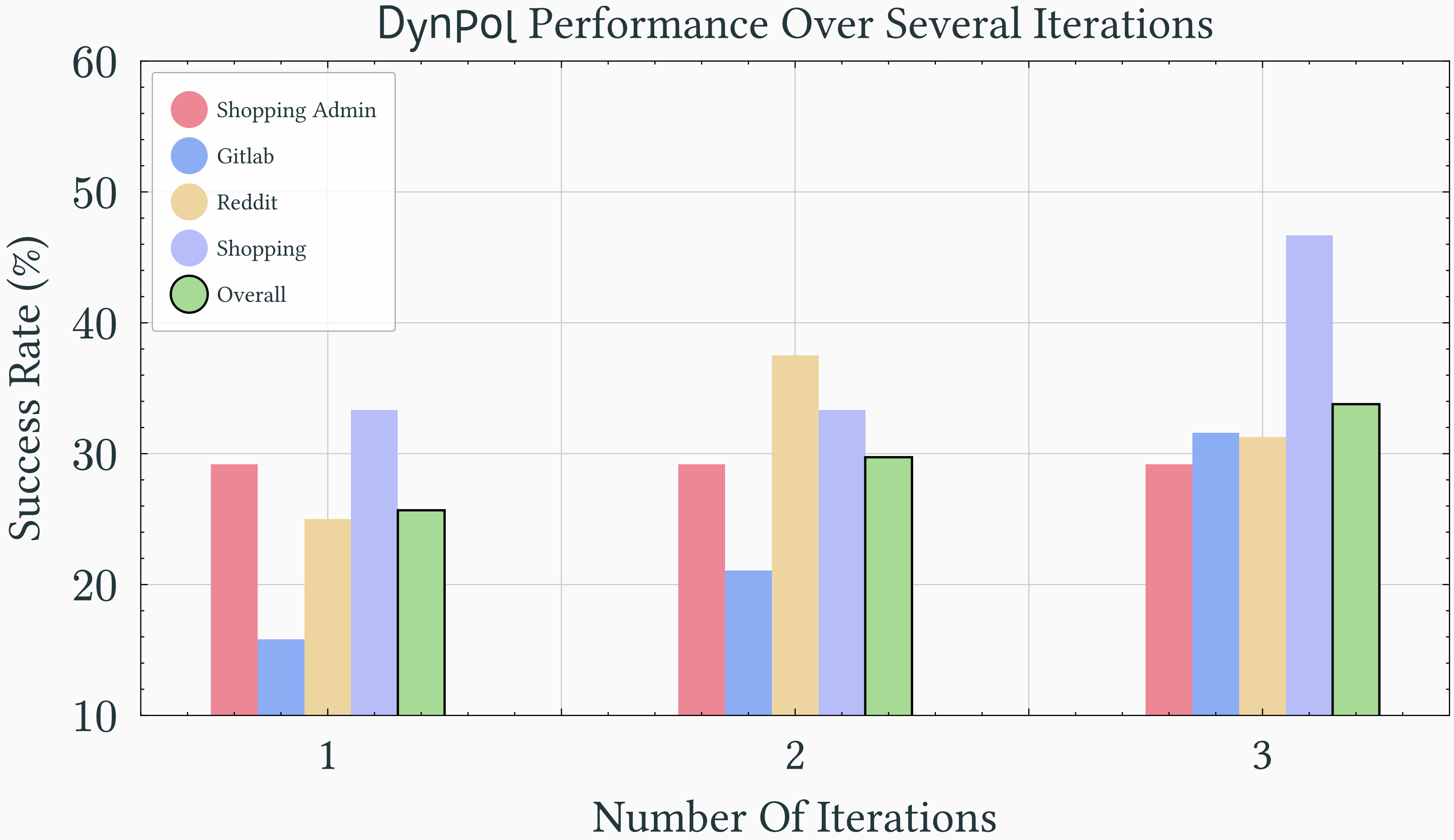
Table 1: WebArena Success Rates for 50 (or 80) randomly sampled tasks for several architectures

4.5 Results Graph

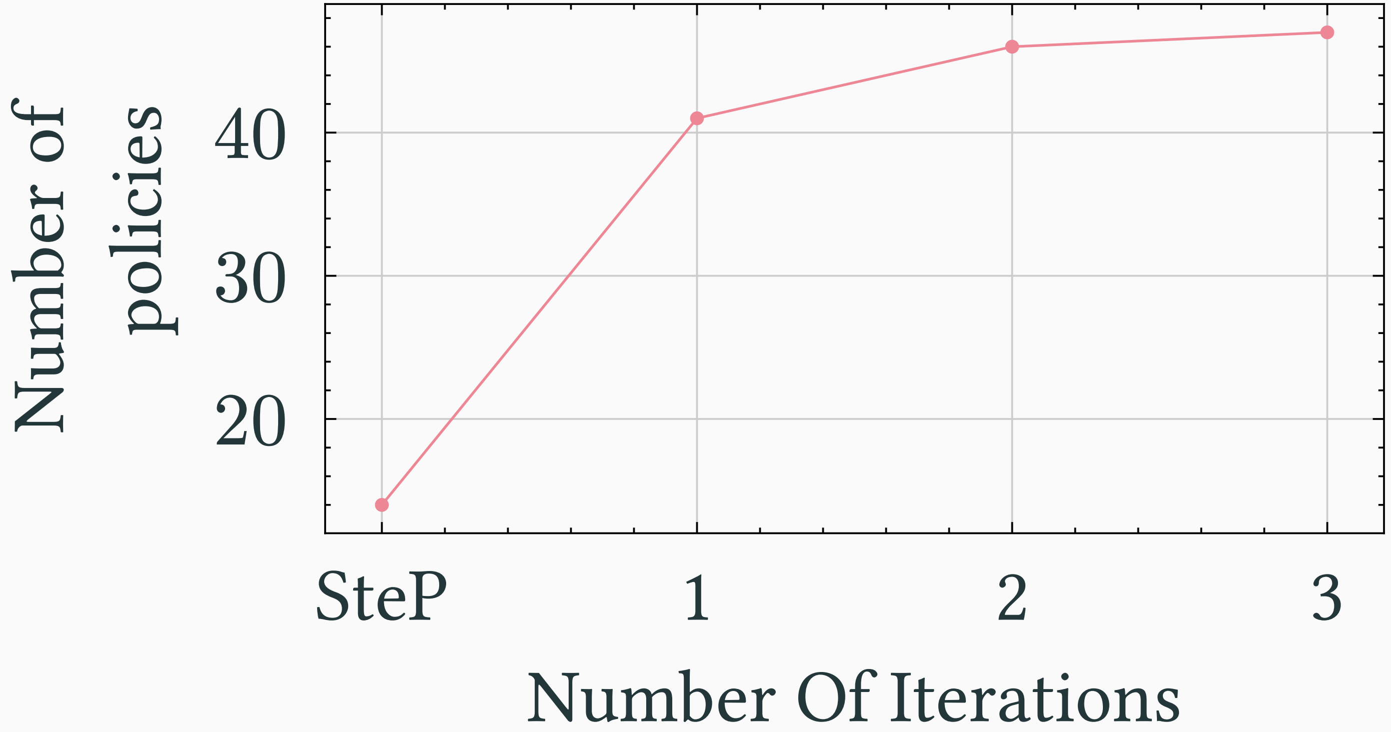


5. Analysis

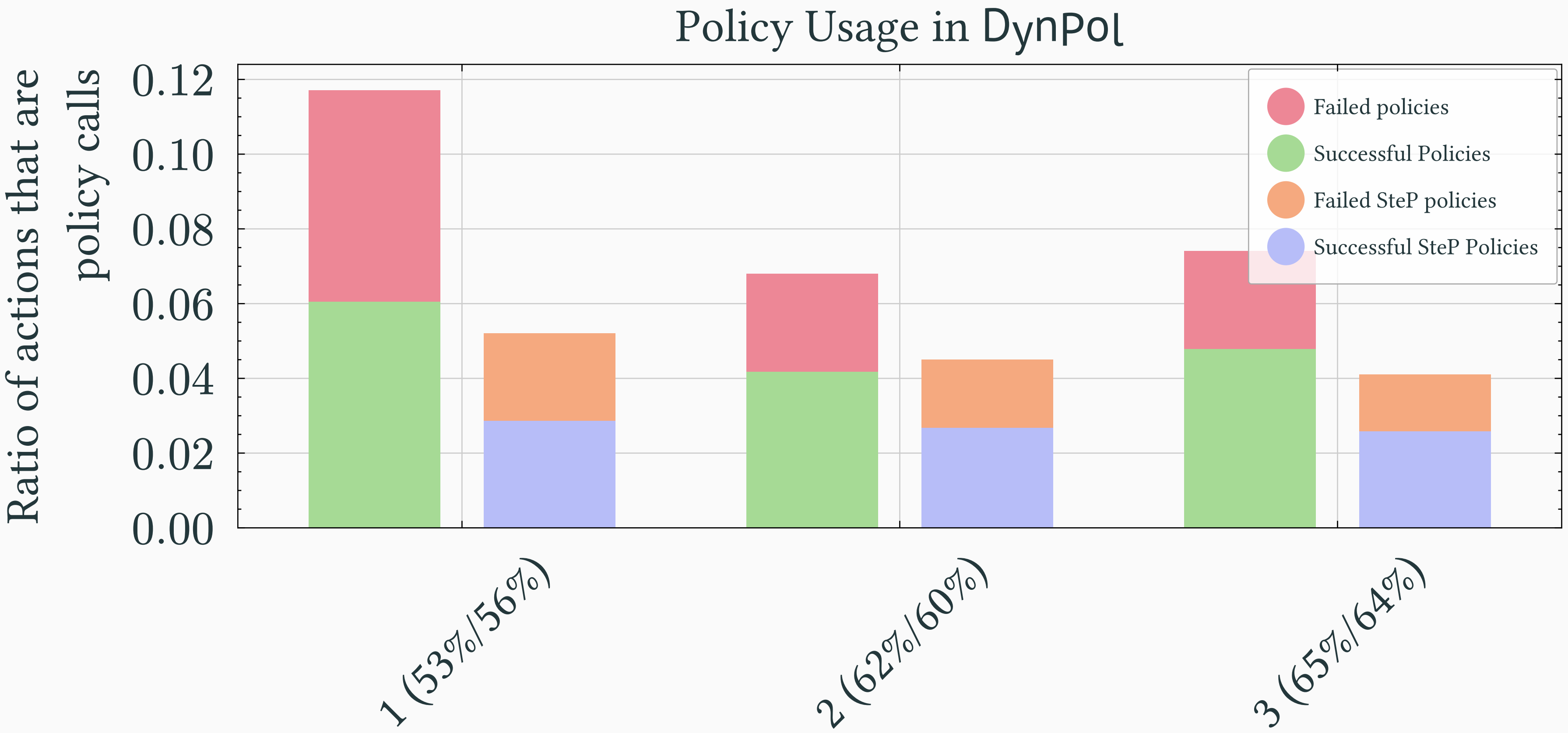
5.1 DynPol over several iterations



Policy Library Size Evolution Over Several Iterations



5.3 Policy Success



Number of iterations alongside the success rates of the policies (Left : All Policies, Right : SteP Policies)

6. Conclusion

6.1 Limitations

- Performance Gap
- Unreliability of LLM-driven prompt optimization
- Catastrophic forgetting in self-improvement
- Limitations of text-only approach
- Time-intensive execution

6.2 Future Works

- More experiments
- Improve prompts
- Add visual components

7. Appendix

Bibliography

[1] P. Sodhi, S. R. K. Branavan, Y. Artzi, and R. McDonald, "SteP: Stacked LLM Policies for Web Actions." [Online]. Available: https://arxiv.org/abs/2310.03720

[2] S. Marreed *et al.*, "Towards Enterprise-Ready Computer Using Generalist Agent." [Online]. Available: https://arxiv.org/abs/2503.01861

[3] S. Zhou *et al.*, "WebArena: A Realistic Web Environment for Building Autonomous Agents." [Online]. Available: https://arxiv.org/abs/2307.13854

[4] H. Su, R. Sun, J. Yoon, P. Yin, T. Yu, and S. Ö. Arık, "Learn-by-interact: A Data-Centric Framework for Self-Adaptive Agents in Realistic Environments." [Online]. Available: https://arxiv.org/abs/2501.10893

[5] K. Yang *et al.*, "AgentOccam: A Simple Yet Strong Baseline for LLM-Based Web Agents." [Online]. Available: https://arxiv.org/abs/2410.13825

Bibliography

[6] S. Murty, H. Zhu, D. Bahdanau, and C. D. Manning, "NNetNav: Unsupervised Learning of Browser Agents Through Environment Interaction in the Wild." [Online]. Available: https://arxiv.org/abs/2410.02907

[7] R. Zhang *et al.*, "Symbiotic Cooperation for Web Agents: Harnessing Complementary Strengths of Large and Small LLMs." [Online]. Available: https://arxiv.org/abs/2502.07942

[8] G. Wang *et al.*, "Voyager: An Open-Ended Embodied Agent with Large Language Models." [Online]. Available: https://arxiv.org/abs/2305.16291

[9] S. Murty, C. Manning, P. Shaw, M. Joshi, and K. Lee, "BAGEL: Bootstrapping Agents by Guiding Exploration with Language." [Online]. Available: https://arxiv.org/abs/2403.08140

[10] T. Khot *et al.*, "Decomposed Prompting: A Modular Approach for Solving Complex Tasks." [Online]. Available: https://arxiv.org/abs/2210.02406

Bibliography

[11] M. Ghasemi and D. Ebrahimi, “Introduction to Reinforcement Learning.” [Online]. Available: https://arxiv.org/abs/2408.07712

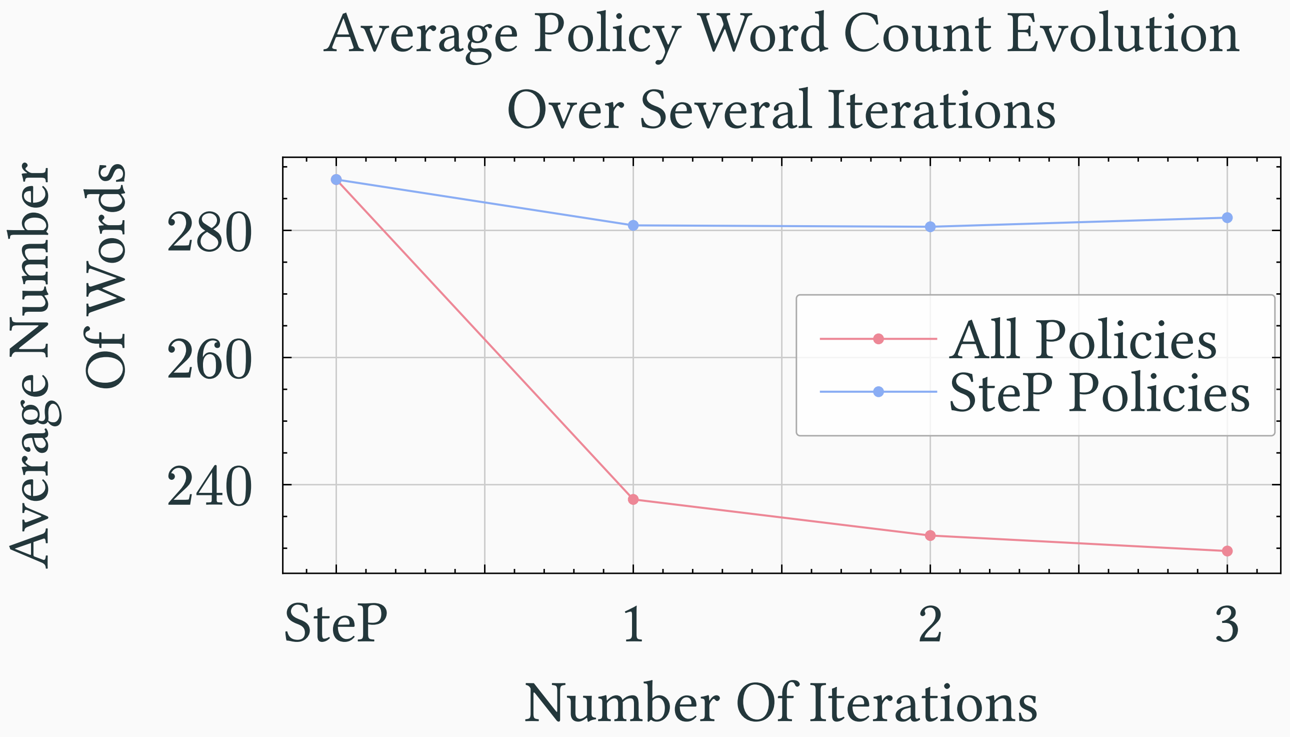
[12] M. Yuksekgonul *et al.*, “TextGrad: Automatic "Differentiation" via Text.” [Online]. Available: https://arxiv.org/abs/2406.07496

[13] Y. Lu, J. Yang, Y. Shen, and A. Awadallah, “OmniParser for Pure Vision Based GUI Agent.” [Online]. Available: https://arxiv.org/abs/2408.00203

[14] S. Yao *et al.*, “ReAct: Synergizing Reasoning and Acting in Language Models.” [Online]. Available: https://arxiv.org/abs/2210.03629

[15] J. Shen *et al.*, “ScribeAgent: Towards Specialized Web Agents Using Production-Scale Workflow Data.” [Online]. Available: https://arxiv.org/abs/2411.15004

7.2 Policy Word Count Evolution



SteP policies word count < DynPol - generated policies word count

SteP policies' instructions contain toy examples

7.3 Pseudo-Code of DynPol Main Loop

Python

```python
def get_action(observation, objective, is_first_step):
    if is_first_step:
        # Initialize the policy stack with a site-dependant policy
        policy_stack = Stack.init(root_policy)
        # Automatic Curriculum, seeing if new policies need to be added to the policy library
        relevant_policies = policy_library.retrieve(objective)
        # LLM Call to choose which policies to use
        chosen_policies = automatic_curriculum(objective, observation, relevant_policies)
        for policy in chosen_policies:
            if not (policy in policy_library):
                policy_library.add(policy)

```

7.3 Pseudo-Code of DynPol Main Loop

```
        current_policy = policy_stack.top()
        relevant_policies = policy_library.retrieve(objective)
        # LLM call to determine which action to take
        action = get_action(objective, observation, current_policy, relevant_policies)

        match action.type:
            case PAGE_OPERATION:
                env.interact(action)
            case POLICY_CALL:
                policy_stack.push(action.policy)
            case STOP:
                finished_policy = policy_stack.pop()
                # Self-Improvement Mechanism
```

7.3 Pseudo-Code of DynPol Main Loop

```
        critique = get_critique(finished_policy.trajectory, objective)
        policy_library.update_usage(critique.is_success)
        if rewrinting_needed(policy_library.get_usage(finished_policy)):
            new_guidance = rewrite(finished_policy.guidance, critique, objective, finished_policy.trajectory)
            policy_library.update(finished_policy, new_guidance)
```

7.4 Sample Policy Prompt

Based on the plan and explanation, the guidance for the web agent to solve the task is:
* To find the latest post related to a specific topic, start by examining the search results page and looking for the most recent timestamp.
* Use the website's search function or filtering capabilities to narrow down the search and highlight the latest post. Look for options to sort posts by timestamp in descending order (newest first).
* If sorting or filtering options are available, use them to highlight the latest post.
* Clearly identify the latest post and its author by looking for the username associated with the post.
* Verify that the highlighted post is indeed the latest one related to the topic by checking the timestamp and ensuring that the post is relevant to the search query.
* Provide clear evidence in the observation that the task has been completed by highlighting the latest post and its author.

Examples:
* If the website has a search box, type in the keyword to find posts related to the topic, and then look for the most recent timestamp to identify the latest post.
* If the website has filtering options, select the option to filter posts by keyword or topic, and then select the topic to find related posts. Look for options to sort posts by timestamp in descending order.
* If the website displays user posts in a list, look for the most recent timestamp to identify the latest post related to the topic, and then extract the author's username.
* If the website displays user posts in a grid, look for the post with the most recent timestamp, extract the author's username, and verify that the post is relevant to the search query.

7.5 Code Repartition

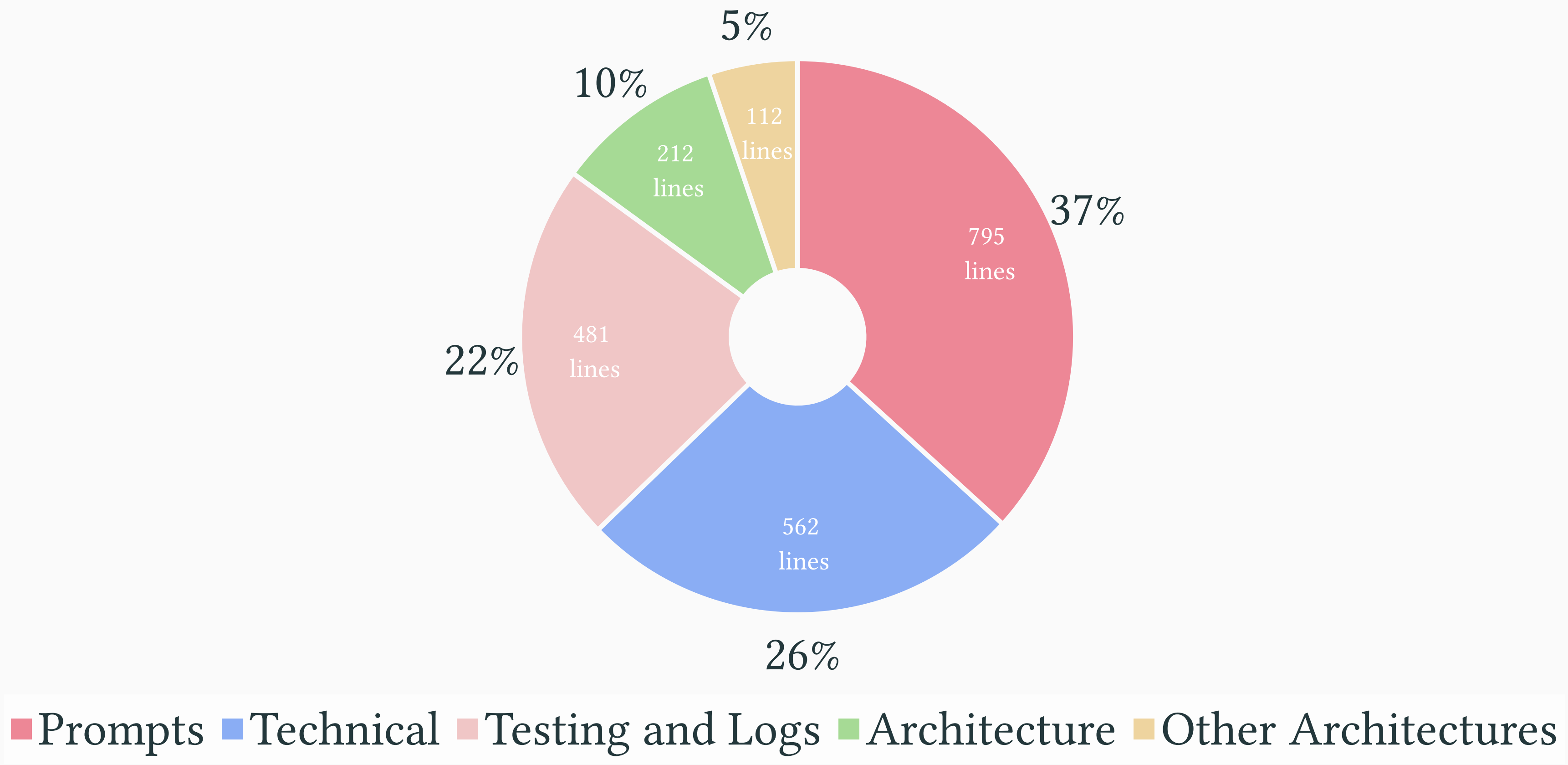


Figure 16: Repartition of the lines inside the codebase

7.6 Time Allocation

March

- Literature for the initial topic
- Switching topic
- Literature for the actual topic
- Initial DynPol draft

April

- Implementing DynPol
- Implementing tools (logging, interacting with web env,...)

May

- Refining DynPol (Automatic Curriculum details, Critique, ...)
- Implementing baseline
- Starting to setup MareNostrum access

7.6 Time Allocation

June

- Experiments
- Refining DynPol
- Set up our LLM on MareNostrum
- Fixing the WebArena AWS instance

July

- More Experiments
- Prompt Engineering on DynPol
- Fixing MareNostrum problems
- Starting report

August

- Final Experiments
- Finishing report