A Survey of the Training Process of LLM-Based Agents

Anonymous ACL submission

Abstract

Autonomous agents based on large language models (LLMs) are becoming an increasingly prevalent paradigm for tackling complex and real-world tasks. Despite the remarkable zero-shot capabilities of modern LLMs, specialized agent training is often essential to obtain reliable and improved performance for specific target tasks. In this survey, we provide a process-centric review of the training of LLM-based agents, with a primary focus on methods that update the parameters of the main agent policy. Rather than organizing prior work only by application domain, agent capability, or learning algorithm, we examine the end-to-end pipeline through which LLM-based agents are trained: environment setup, data and trajectory construction, learning-signal design, and optimization objectives and schemes. This perspective highlights how design choices at different stages interact with one another. Finally, we conclude with discussions on potential future directions.

1 Introduction

Autonomous agents based on large language models (LLMs) have been applied to a wide variety of real-world applications (Wang et al., 2024a; Xi et al., 2025b; Luo et al., 2025a). These agents utilize the LLMs as the core cognitive engine to process environmental perceptions and decide subsequent actions. This paradigm has shown great promise for the development of AI assistants that can automate workflows across diverse domains.

Earlier LLM-based agent systems primarily depend on prompt engineering and in-context learning applied to the underlying LLMs (Hong et al., 2023; Qian et al., 2024; Wu et al., 2024). Although such methods provide a flexible and lightweight way to construct agents, they are often insufficient for complex or specialized tasks. A key challenge is that agent tasks require the model to act within an external environment rather than only generate a textual response. In practice, the LLMs may be unfamiliar with the target environmental observations or action spaces, which are not adequately represented in their pre-training or standard post-training stages. These requirements are not always adequately covered by standard LLM pre-training or generic post-training. This motivates the need for dedicated *agent training*, which could fundamentally adapt these LLMs to effectively handle the target agent tasks.

Algorithm 1 A simplified agent training procedure.

Input: Environment `E`, Agent System `A`, Underlying Large Language Model `M`, Training Iteration `I`.
Output: Trained Model `M'`.

```
M' = M                                  ▷ Start with an Initial Model
for iter in range(I) do
    E.setup()                           ▷ Setup Environment (§3)
    Q = collect_query()                 ▷ Obtain Query (§4.1)
    T = sample(Q, A, M', E)             ▷ Sample Trajectory (§4.2)
    S = get_signal(Q, T, E)             ▷ Learning Signal (§5.1)
    M' = train(Q, T, S, M')             ▷ Model Training (§5.2)
end for                                 ▷ Training Scheme (§5.3)
return M'
```

Although there have been many existing surveys for LLM-based agents, few of them place a specific and central focus on the training process of LLM-based agents. General surveys cover agent research *in a broad manner* (Wang et al., 2024a; Xi et al., 2025b; Luo et al., 2025a) or focus on *generalized improvement strategies* (Gao et al., 2025a; Fang et al., 2025a; Du et al., 2026), while others focus on *specific aspects*, such as memory (Zhang et al., 2025k), planning (Huang et al., 2024), tool-use (Qin et al., 2024; Qu et al., 2025a), multi-agent (Guo et al., 2024; Tran et al., 2025) and evaluation (Yehudai et al., 2025), *specific algorithms*, such as reinforcement learning (Zhang et al., 2025d), as well as *specific applications*, such as multi-modal (Xie et al., 2025b), web (Ning et al., 2025) and GUI (Zhang et al., 2025b; Tang et al., 2025a; Nguyen et al., 2025a) agents. These surveys provide valuable perspectives, but they typically or-

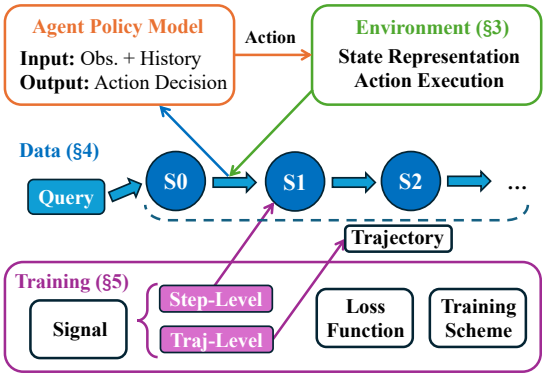

Figure 1: Overview of this survey. The pipeline consists of environment setup, data and trajectory construction, and training design. Compared with standard LLM post-training over static input-output pairs, agent training is distinguished by environment interaction, trajectory dependence, and long-horizon credit assignment.

ganize the literature by agent capability, application domain, or training algorithms. In contrast, this survey focuses on the end-to-end process through which a general-purpose LLM is adapted into an environment-interactive agent policy.

The primary highlight of this work is its *process-centric organization*. Instead of a purely taxonomic approach, this work is structured around the steps of a typical agent training procedure, including key topics such as environment setup (§3), data collection (§4), and learning signal generation (§5). We outline a typical agent training process in Algorithm 1, and the remainder of this survey is structured to examine each of the main steps in this process. With this practical organization, we hope to provide actionable guidance and clear design choices for better implementing and optimizing the entire agent training process.[1]

2 Agent Training vs. LLM Training

Although LLM-based agent training often adopts objectives from general LLM post-training, the underlying training problem is structurally different. General LLM training usually optimizes responses to relatively static prompts or dialogue contexts, but agent training optimizes an environment-interactive policy. Given environmental observations and interaction history, the agent needs to produce executable actions that could change future states and observations. This makes the learning unit a trajectory of action-state pairs rather than an isolated input-output pair, introducing additional challenges such as environmental grounding, trajectory collection, and long-horizon credit assignment.

Figure 1 provides an overview of this survey and illustrates the distinctions. The key point is that ordinary training objectives become agent-specific once they are coupled with environment design, trajectory construction, and feedback assignment. These aspects must be carefully analyzed and designed in agent training, which are crucial for the effectiveness of the training procedure. Therefore, this survey studies agent training as a process that connects environments, data, learning signals and schemes, rather than as a direct application of standard LLM post-training.

3 Environment

The environment is a key component that distinguishes agent tasks from standard non-interactive text generation tasks. An agent operates in a multi-turn manner by iteratively receiving observations from the environment and deciding actions that further influence the environment's state. Consequently, the representations of the observations and actions directly define the input and output spaces of the underlying decision model. A key motivation for agent training is precisely the LLM's unfamiliarity with these specific environmental settings.

The nature of the environment is inherently tied to the target task, with each task type putting distinct specifications. For example:

- **Search agents** (Xi et al., 2025a; Zhang et al., 2025i) typically interact with external retrievers or search engines, where actions are formalized as search queries and observations are the corresponding retrieved texts.
- **Web agents** (Ning et al., 2025) interact with a browser. They receive observations of web pages, and their actions include typical browser operations like clicking, scrolling, and typing.
- The scope can be extended to **GUI applications** (Zhang et al., 2025b) for general computer use (Hu et al., 2025b) if visual inputs of screenshots are incorporated.
- In **software engineering** tasks (Liu et al., 2024; Wang et al., 2025d), the environment includes a code repository, enabling agents to apply actions such as code editing and repository manipulation.

[1]The detailed paper collection procedure is described in Appendix A. In this work, we primarily focus on approaches that involve parameter updates to the main agent policy, while other related aspects are discussed in Appendix B. We also provide agent backgrounds in Appendix D as well as summary and guidance tables in Appendix E.

3.1 Observation Representation

Observation Processing. The current state's observation from the environment is a crucial input signal that directly influences the agent's decision. Compared to general natural language inputs, a unique property of agent observations is their tendency to be *highly structured and complex*. This complexity often requires specialized processing steps to enable the model to accurately understand the current state. To process input observations:

- **Augmentation** methods enrich the original raw observations to ease understanding. For example, techniques like Set-of-Mark (Yang et al., 2023), which adds special marks to annotate visual elements, have been utilized in agent applications (Zhang et al., 2025c). SeeAct (Zheng et al., 2024a) enhances element grounding by augmenting inputs with element attributes and textual choices. OmniParser (Lu et al., 2024b) specifically fine-tunes models to parse screenshots and provide function semantic annotations.

- **Simplification** methods are essential because raw observations are often complex and redundant, a concern amplified by the context length constraints of LLMs. Synapse (Zheng et al., 2024b) extracts task-relevant information from raw states to form cleaner observations. Similarly, AgentOccam (Yang et al., 2025a) demonstrates that providing less redundant yet informative observations can improve agent performance. With similar objectives, LCoW (Lee et al., 2025) trains a contextualization module to enhance this process, FocusAgent (Kerboua et al., 2025) uses a lightweight retriever to extract the most relevant lines from the observation, and Prune4Web (Zhang et al., 2025f) generates executable Python scoring programs to dynamically filter elements in observations.

History Processing. In addition to the current observation, the record of past interaction steps is a critical input source for the next action decision. A primary challenge here is that as the number of interaction steps increases, the cumulative history representation can quickly grow, making it inefficient and difficult for the LLM to process the entire context effectively. To manage the growing history:

- **Truncation** provides a straightforward solution, by adopting a local window that includes only the details of the most recent steps and omitting previous steps (Tang et al., 2025b).

- **Summarization** provides a more informative approach by converting the previous trajectory into a more concise representation. This approach can be applied periodically (Yen et al., 2025), triggered by heuristic rules such as approaching the token budget (Wu et al., 2025e; Lu et al., 2025b), or dynamically decided based by the agent itself (Sun et al., 2025c; Ye et al., 2025).

The history management directly relates to the broader concept of agent memory (Zhang et al., 2025k) and context engineering (Mei et al., 2025), as the previous trajectory can be regarded as the agent's short-term memory, necessitating efficient encoding and retrieval to optimize the sequential decision process.

3.2 Action Representation

The output generated by the LLM directly decides the next action. For LLM-based agents, the output space is *inherently diverse and structured*. This is one of the key differences to classical agents, which often operate with a few fixed action options. The output decision is typically formalized in the style of **function calling** (Patil et al., 2024), where the LLM is required to select and parameterize a function from either a pre-defined action list or a tool-use library. To facilitate reliable parsing and execution, these output representations often utilize structured text formats such as JSON or other special formats. Moreover, **code-based representation** is a natural and elegant way to represent the output (Wang et al., 2024d), and it can enable flexible action compositions by generating and executing programs (Nguyen et al., 2025b). Beyond the format, the **design of the action space** is a critical area of investigation, for which there have been some recent discussions on building more agent-friendly APIs and tools (Song et al., 2025c; Lù et al., 2025; Zhang et al., 2025a).

Takeaway for §3: Environment design fundamentally shapes agent training by defining the observation, history, and action representations, often requiring specialized processing.

4 Data

Data quality is one of the most important factors for agent training. While **manual annotation** could offer the best way to ensure quality (Lu et al., 2024a; Rawles et al., 2023; Deng et al., 2023; Li et al., 2024), it is inherently tedious and cost-prohibitive.

Moreover, agent tasks require not only the input queries, but also the multi-step interaction trajectories, posing great challenges to the data curation process. Consequently, recent research has focused on automated and scalable LLM-based **data synthesis** approaches (Tan et al., 2024). The following subsections[2] will illustrate how these techniques are specifically applied to agent tasks.

4.1 Queries

Similarly to standard LLM tasks, agent training first requires the synthesis of queries that define the goal that the agent needs to complete. We categorize related methods based on the complexity and the grounding level of the synthesizing process.

Direct Synthesis. This approach uses the inherent instruction-following capabilities of LLMs to synthesize new task queries in a direct way. The simplest method involves **direct prompting**, where an LLM is asked to generate target tasks, often inspired by existing examples. This is a technique widely adopted for agent task synthesis, drawing inspiration from Self-Instruct (Wang et al., 2023; Patel et al., 2024; He et al., 2025). Considering that agent tasks are intrinsically grounded in a specific environment, a natural and necessary extension is **context-aware querying**. Here, the synthesizing LLM is provided with environmental context, such as website names and screenshots (Zhou et al., 2025), crawled web pages (Wu et al., 2025b), pre-training documents (Cen et al., 2025), or multi-hop paths in knowledge graphs (Lu et al., 2025c). However, queries constructed via these direct methods may still suffer from limitations, potentially being too simple or lacking intricate connections to the target environment's full complexity.

Iterative Extension. To address the simplicity limitation of previous methods and improve the overall difficulty of the synthesized queries, an **iterative extension** approach can be adopted. This technique usually starts with a seed item and incrementally transforms it into a more complex goal through repeated refinement. The core goal is to generate *challenging but realistic* query-answer pairs, a process heavily investigated in recent work of deep search agents (Shi et al., 2025a; Gao et al., 2025b; Li et al., 2025b; Tao et al., 2025). The extension process itself is typically mediated by an LLM, often *equipped with external tools*. For example, WebExplorer (Liu et al., 2025b) operates by prompting LLMs equipped with search and browsing tools to guide the construction. Moreover, this approach can be used to build more complex agent tasks; AgentSynth (Xie et al., 2025a), for example, iteratively forms a sequence of subtasks that are then summarized into a composite task for computer-use agents. Nevertheless, this approach may produce unnatural queries, and building complex yet natural tasks remains a challenge.

Exploratory Construction. To obtain queries that are more connected to the target environments, the construction itself can be viewed as an agent task, and a dedicated data construction agent can be designed to explore the environment. This typically follows a **reverse construction** process: firstly interacting with the environment with an exploration-based policy and then reversely synthesizing the query with the exploration trajectory. This approach has been adopted for a variety of target environments and tasks, including search (Wang et al., 2025c), web (Murty et al., 2024b; Pahuja et al., 2025; Trabucco et al., 2025), GUI (Su et al., 2025a; Sun et al., 2025b; Ramrakhya et al., 2025), and tool-use (Zhai et al., 2025). Although this method can create queries that are closely related to the target environment, there still remain challenges such as constructing non-trivial and challenging tasks and controlling the extra exploration cost.

4.2 Trajectories

In addition to the task queries, agent training also requires the interaction trajectories to support the full multi-turn decision process of the agent.

Trajectory Conversion. A simple way to collect trajectories is through **trajectory conversion**, by repurposing and transforming existing trajectories originally collected for related tasks (Gandhi et al., 2024). For agent applications, this may be challenging due to the format differences across various target tasks and environments. This complexity has driven standardization efforts, with work such as ADP (Song et al., 2025b) and AgentOhana (Zhang et al., 2024) aiming to define *unified data formats and protocols* to facilitate better data reuse. Furthermore, when the format discrepancy between the original and desired data is large, **LLM-based transformation** can be utilized, where an LLM acts as a robust parser and translator to complete the conversion (Yin et al., 2024a). While such ap-

[2] It is also important to perform *data filtering* to ensure the quality of the training data. This is closely tied to the construction of training signals, which will be covered in §5.1.

proaches effectively utilize existing resources, the availability of relevant and high-quality source data is often lacking, necessitating more target-specific trajectory construction approaches.

Sampling. The most fundamental method to collect agent trajectories is **sampling**, which involves using a designated policy to run target tasks directly within the corresponding environment. For the underlying LLM policy used during the sampling process, there are two primary choices: utilizing a stronger teacher model to generate high-quality data that guides the learner (Chen et al., 2023; Zeng et al., 2024), or employing the current version of the agent model itself to enable direct self-improvement (Patel et al., 2024; Fang et al., 2025b). While the simplest implementation involves independently sampling each trajectory in parallel, more complex algorithms like **tree search** (Zhou et al., 2024a; Koh et al., 2025; Lin et al., 2025a; Hou et al., 2025) can be employed to perform deeper exploration and generate better paths.

Special Construction. There have also been a range of **specialized approaches** developed to construct high-quality agent trajectories by leveraging diverse external knowledge or employing special refinements. Some methods focus on **knowledge-based transformation**, such as Synatra (Ou et al., 2024), which uses procedural knowledge resources like WikiHow as a source to be transformed into trajectories, and AgentTrek (Xu et al., 2025), which crawls internet tutorials and converts them into structured representations suitable for guided replay. Other techniques center on refinement and bootstrapping for **quality improvement**: BAGEL (Murty et al., 2024a) applies a round-trip bootstrapping approach that transforms between synthetic instructions and refined trajectories, while UI-Simulator (Wang et al., 2025e) employs a trajectory wrapper that reconstructs raw rollouts into high-quality training instances by inferring user instructions and generating a coherent reasoning process. Finally, methods like ActRe (Yang et al., 2024) focus on **reasoning injection** by randomly sampling an action and subsequently generating a detailed explanation to compose a full reason-then-act trajectory, and WebCoT (Hu et al., 2025a) curates trajectory data that exemplifies special abilities by reconstructing the agent's reasoning algorithms into chain-of-thought rationales. These specialized techniques enable the creation of richly annotated agent trajectories.

Takeaway for §4: Agent training requires scalable synthesis of both task queries and interaction trajectories, with recent methods using LLM-driven generation, environment-grounded exploration, and sampling-based rollouts to construct diverse and high-quality training data.

5 Training

5.1 Signal

The acquisition and utilization of high-quality learning signals is one of the most important factors for successful agent training. In this sub-section, we examine learning signals from three complementary perspectives: **source** (where the signals originate), **granularity** (the level at which the signals are provided) and **form** (the representation of the signals themselves).

5.1.1 Signal Source

Expert Supervision. The most straightforward way to obtain training signals relies on the direct **expert supervision**, including manual annotation or knowledge distillation from superior "teacher" models (Hinton et al., 2015; Kim and Rush, 2016). One typical way is to collect *oracle trajectories* using a teacher model and use them as direct demonstrations to train the agent policy (Chen et al., 2023; Zeng et al., 2024).

Environment-Based Feedback. In many agent scenarios, training supervision is provided by pre-defined external signals from environments. LLM-based agent tasks differ fundamentally from plain LLM tasks in that the agents are interacting with the environment, which can directly provide **environmental learning signals**. The precise form of such signals depends on the target environment: for example, they may be directly determined by changes in the environmental state (Brockman et al., 2016; Shridhar et al., 2021) or calculated by pre-defined evaluation functions (Yao et al., 2022; Zhou et al., 2024b). A typical and valuable characteristic of these external signals is that they could be **rule-based or verifiable**, making them suitable for methods like reinforcement learning with verifiable rewards (RLVR) approaches (Lambert et al., 2025; Guo et al., 2025a). If the target task is to provide a single final answer, signals can be readily obtained by matching the predicted answer with the gold reference answer (Jin et al., 2025; Song et al., 2025a; Chen et al., 2025; Qian et al., 2025;

Wei et al., 2025). In addition to final answers, there can also be external signals that provide **partial evaluation** of the predicted trajectory, such as entity matching ratios (Zhao et al., 2025) or sub-goal scaffolding (Luo et al., 2025b) in complex search tasks. An interesting connection is that external signals can often be obtained **by construction** during the data synthesis phase (§4), where the queries and the corresponding target answers or evaluation functions are constructed simultaneously.

Model-Based Feedback. The learning signals can also be generated by models. It is common to use stronger models in the way of LLM-as-a-judge (Gu et al., 2024), such as providing judgments as RL rewards (Lee et al., 2024) or filtering highest-quality data (Zeng et al., 2024; Trabucco et al., 2025). Moreover, the agent or its underlying LLM to be trained can also serve as an internal source, generating its own learning signals. One of the key metrics is **confidence**, which can potentially indicate the correctness of generated trajectories. This is typically calculated by checking the generation probability of the model outputs (Wu et al., 2025d) or by verifying consistency across multiple generation paths (self-consistency; Huang et al., 2023; Kang et al., 2025). Leveraging the LLM's general-purpose ability to reason and evaluate, the model can be treated as its own judge in a **self-rewarding** setting (Yuan et al., 2024), generating reward signals based on its own assessment of the generated output. Furthermore, LLMs have demonstrated strong abilities of **self-reflection** (Shinn et al., 2023; Madaan et al., 2023), and this inherent capacity can be utilized to provide internal critique (Zhang et al., 2025g; Li et al., 2025c).

5.1.2 Signal Granularity

Agent tasks are unique in that they consist of multi-turn trajectories containing interleaved action predictions and environmental observations, rather than just a linear sequence of generated tokens as in plain LLM tasks. Consequently, learning signals can be provided at various levels, including trajectory-level, step-level, or even token-level. Considering the multi-step characteristic of agent tasks, the most interesting discussions are on the comparison between *trajectory-level versus step-level signals*. This directly mirrors the general LLM alignment debate over outcome reward models (ORM) versus process reward models (PRM). While ORM signals are typically easier to obtain, PRM signals offer denser supervision (Uesato et al., 2022; Lightman et al., 2024; Zheng et al., 2025a).

In agent tasks, ORM provides **trajectory-level signals**, typically as one aggregated evaluation score for the full agent trajectory. Many of the verifiable rewards discussed in the previous section (§5.1.1) are examples of this granularity. The inherent challenge of this approach is that such signals are sparse, often provided only after the full trajectory is completed, which may lack sufficient supervision to effectively guide the learning for the intermediate decision steps.

Considering **step-level signals** is a natural extension, as each step in the agent trajectory corresponds directly to one action decision. Nevertheless, these finer-grained signals are much more difficult to obtain, for which manual annotation (Chae et al., 2025b) could be tedious and expensive. We categorize the typical automatic approaches used to acquire step-level signals in the following:

- **Critic-based** methods use LLM-based reward models, where an LLM critic can directly examine each step's details and assign rewards (Tan et al., 2025b; Yu et al., 2025a; Zhang et al., 2025j). In addition, specific critic models can be explicitly trained to provide these signals. For instance, Liu et al. (2025d) optimize an implicit PRM to provide step-level rewards as the log ratios between the action probabilities with the learned PRM and the old policy, while Xi et al. (2025c) adopt a temporal difference-based estimation method for PRM training. Similarly, Zhou et al. (2024c) learns a high-level value function with off-policy RL to provide signals for low-level action generation.
- **Search-based** techniques use complex search algorithms to yield more fine-grained rewards. A common strategy involves generating a search tree with the current policy and assigning rewards to the intermediate steps according to the rewards of its descendants (Hou et al., 2025). The search tree can be constructed in various ways, including Monte Carlo sampling (Xiong et al., 2024), entropy-guided rollout (Shen et al., 2025), or iterative expansion (Lin et al., 2025b; Wu et al., 2025a). Additionally, if step states across different trajectories can be grouped, step-level signals can be calculated using the state rewards within each group (Feng et al., 2025).
- **Task-specific** methods provide step-level signals uniquely tailored to target tasks. Examples

include rewards based on information gain by checking action probabilities depending on the ground-truth answer (Wang et al., 2025a), rewards based on the usefulness and redundancies of retrieved documents for each query action (Zheng et al., 2025c), and specialized rewards generated by detecting suboptimal behaviors for the target search task (Wu et al., 2025c).

Although step-level signals provide denser learning supervision, the performance and stability could be highly dependent on the choice of the learning algorithms (Wang and Ammanabrolu, 2025), and the potential noise inherent in such signals should be carefully considered.

5.1.3 Signal Form

The learning signals can be presented in various forms. The most common form is **numeric rewards**, which provide a scalar judgment on the correctness or goodness of the evaluated trajectory or action. In addition, **natural-language feedback** provides more detailed critique, providing richer and qualitative signals for the training of the LLMs (Scheurer et al., 2022; Chen et al., 2024; Xu et al., 2024) and the agents (Yang et al., 2025b). In scenarios where assigning absolute scores is difficult, it might be easier to provide **preference signals**, indicating which of two actions or trajectories is superior. Such comparative signals are utilized to perform preference learning (Rafailov et al., 2023) for agents (Xiong et al., 2024; Song et al., 2024). Moreover, if expert sources can provide **direct demonstrations** of the correct actions, these signals can be directly utilized for imitation learning.

A special type of learning signal is the supervision of **error correction**. For example, Lyu et al. (2025) propose a student-centered framework where an expert corrects only the earliest error in a student-generated trajectory, iteratively guiding the erroneous path towards a correct one. However, a central difficulty in obtaining correction signals is in the problem of **error attribution**, which remains an important research question (Cemri et al., 2025; Zhang et al., 2025h,e; Zhu et al., 2025).

5.2 Loss

The loss functions utilized for agent training largely stem from those established for plain LLM training. The most widely used losses include the standard cross-entropy loss in supervised fine-tuning (SFT), the preference loss used in direct preference optimization (DPO; Rafailov et al., 2023), and various reinforcement learning (RL) losses (Sutton and Barto, 2018). The fundamental details of these functions are omitted here as they are well-known, and unfamiliar readers are referred to related surveys (Zhao et al., 2023; Wang et al., 2024c; Tie et al., 2025; Liu et al., 2025c; Du et al., 2026; Zhang et al., 2025d). Each loss function requires a corresponding type of learning signal (§5.1.3): SFT needs direct demonstrations of the oracle action, DPO requires preference pairs of positive and negative action instances, and RL demands rewards associated with the agent trajectories.

For agent applications, there are several special considerations for the loss. First, since agent trajectories consist of interleaved agent actions and environmental observations, it is common to adopt **observation masking** (or loss masking) during training, as the observation tokens are not directly generated by the model. Jin et al. (2025) show that leveraging loss masking for retrieved tokens can be beneficial. Second, there have been various **modifications** of standard RL losses to adapt to the agent setting and further stabilize training (Wang et al., 2025f; Yu et al., 2025b; Deng et al., 2025; Li et al., 2025a). Finally, considering the multi-turn nature of agent trajectories, different loss functions can be applied at different levels, exemplified by ArCHer (Zhou et al., 2024c), which adopts a **hierarchical approach** running two distinct RL algorithms in parallel to manage high-level action decision and low-level action token generation.

5.3 Scheme

In Algorithm 1, we only list a simplified version of the agent training scheme; in practice, there are numerous variations with different considerations.

Training Efficiency. In agent training, data collection can be costly, especially in online learning scenarios where trajectory sampling and model updating are interleaved in a fine-grained manner. This poses significant challenges for practical training efficiency. One typical strategy to mitigate this cost is to shift towards **offline learning** approaches (Levine et al., 2020). While pure offline learning assumes training with previously collected data and no additional online interactions, intermediate methods are often considered, adopting an iterative and **coarse-grained** approach for balancing data freshness with computational savings (Patel et al., 2024; Aksitov et al., 2024; He et al., 2025). Furthermore, to fully utilize available computational re-

sources and parallelize operations, **asynchronous training** methods have been increasingly employed (Tan et al., 2025a; Gao et al., 2025b; Jiang et al., 2025; Lu et al., 2025a).

Training Stages. Similar to plain LLM post-training, the agent training procedure can be split into **multiple stages** with different training schemes. A widely adopted scheme involves first performing an offline phase of SFT on expert demonstrations, followed by an online phase of RL to refine the policy through interaction (Vattikonda et al., 2025; Wu et al., 2025b). Furthermore, to enhance the model's foundational understanding and better support agentic behaviors, some work incorporates a **continual pre-training** stage that precedes the main post-training (SFT/RL) fine-tuning (Wang et al., 2025b; Su et al., 2025b; Copet et al., 2025). This additional stage focuses on preparing the model with relevant knowledge and capabilities for effective downstream agent operation.

Training Curriculum. The data and environmental setup can also be adjusted across the training procedure, naturally adopting the idea of **curriculum learning** (Bengio et al., 2009). This approach structures the learning process by starting with simpler tasks or data and gradually increasing the difficulty. Examples of curriculum learning in agent training include starting with easy samples and then progressively introducing more complex ones (Lai et al., 2024), generating new tasks from previous unsuccessful attempts to create a self-evolving curriculum (Qi et al., 2025), and incrementally degrading the quality of simulated environment observations using a curriculum-based rollout strategy to improve generalization (Sun et al., 2025a).

Takeaway for §5: Agent training is driven by learning signals from experts, environments, or models, whose granularity and form determine the training losses, while training schemes organize data collection, optimization, and curriculum for efficient and effective learning.

6 Conclusions and Future Directions

This work presents a structured survey of main-policy training methodologies for LLM-based agents, organizing existing approaches along the key stages of the training pipeline: environment design, data synthesis, and training mechanisms. By examining these components from a process-centric perspective, we aim to clarify the design space of agent training and provide guidance for future research.

Scalable Supervision. One main difficulty in training LLM-based agents is the lack of sufficient high-quality supervision signals. Agent tasks often require long interaction trajectories in complex environments, making data collection costly while reward signals remain sparse or noisy. One promising direction is reducing interaction cost through simulation, where learned environment models allow agents to practice and generate trajectories internally (Ding et al., 2025; Liu et al., 2025a; Team et al., 2025). Another important direction is to construct more reliable automatic feedback mechanisms. For open-ended tasks without clear verifiers, reward models and LLM-based evaluators remain useful but can introduce bias, inconsistency, or reward hacking (Leike et al., 2018).

Multi-modal Agent Training. Many real-world agent applications involve multi-modal environments such as graphical interfaces, visual observations, or audio inputs (Xie et al., 2025b). While the overall training procedure remains similar, these settings introduce additional challenges in grounding and perception, requiring agents to correctly associate language and symbolic reasoning with sensory inputs (Zheng et al., 2024a). Despite recent advances in multi-modal LLMs (Yin et al., 2024b), integrating perception, reasoning, and action into a unified and reliably trained agent system remains an open challenge.

Life-Long Learning. A long-term goal for LLM-based agents is to enable fast and robust adaptation to the dynamic environments encountered in real-world applications (Zheng et al., 2025b). To achieve this, it will be crucial to integrate agent training with established techniques from life-long or continual learning (Chen and Liu, 2018; De Lange et al., 2021; Wang et al., 2024b; Shi et al., 2024). This inherently involves a holistic integration of the different components within the agent system, making it crucial to accurately identify which components should remain invariant across tasks and which should be adapted quickly in response to external environmental changes. More broadly, the community also needs reproducible infrastructure for agent training, which is essential for better comparisons of training approaches.

Limitations

This work has several limitations in its design and scope. Firstly, we specifically focus on the training process of LLM-based agents, omitting other significant aspects such as architectures, evaluation, and specific applications. This strategic choice is made because many of these areas are already well-covered by existing literature, and we refer readers to the other comprehensive agent surveys mentioned in the introduction (§1). Secondly, the descriptions provided in this work are mostly brief to provide a comprehensive coverage within the constraints of page limits. Our approach is to present related works in meaningful and structured groups rather than describing them in unstructured detail. We hope that this work can serve as a high-level index where further details can be found in the corresponding cited papers. Finally, this work is a pure survey without any experiments or empirical results. While performing comparative experiments across various agent training strategies would undoubtedly provide more meaningful and actionable guidance, we leave this resource-intensive effort for dedicated future work.

References

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, Manzil Zaheer, Felix Yu, and Sanjiv Kumar. 2024. ReST meets react: Self-improvement for multi-step reasoning LLM agent. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Stefano V Albrecht, Filippos Christianos, and Lukas Schäfer. 2024. *Multi-agent reinforcement learning: Foundations and modern approaches*. MIT Press.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. Why do multi-agent LLM systems fail? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhepeng Cen, Haolin Chen, Shiyu Wang, Zuxin Liu, Zhiwei Liu, Ding Zhao, Silvio Savarese, Caiming Xiong, Huan Wang, and Weiran Yao. 2025. Webscale-rl: Automated data pipeline for scaling rl data to pretraining levels. *arXiv preprint arXiv:2510.06499*.

Hyungjoo Chae, Namyoung Kim, Kai Tzu iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. 2025a. Web agents with world models: Learning and leveraging environment dynamics in web navigation. In *The Thirteenth International Conference on Learning Representations*.

Hyungjoo Chae, Sunghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, Dongwook Choi, Minseok Kang, Gwanhoon Im, ByeongUng Cho, Hyojun Kim, Jun Hee Han, Taeyoon Kwon, Minju Kim, Beong woo Kwak, and 2 others. 2025b. Web-shepherd: Advancing PRMs for reinforcing web agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Angelica Chen, Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Samuel R. Bowman, Kyunghyun Cho, and Ethan Perez. 2024. Learning from natural language feedback. *Transactions on Machine Learning Research*.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Chenzheng Zhu, Haofen Wang, Jeff Pan, Wen Zhang, Huajun Chen, Fan Yang, and 1 others. 2025. Learning to reason with search for llms via reinforcement learning. *Advances in Neural Information Processing Systems*, 38:85287–85307.

Zhiyuan Chen and Bing Liu. 2018. *Lifelong machine learning*. Morgan & Claypool Publishers.

Jade Copet, Quentin Carbonneaux, Gal Cohen, Jonas Gehring, Jacob Kahn, Jannik Kossen, Felix Kreuk, Emily McMilin, Michel Meyer, Yuxiang Wei, and 1 others. 2025. Cwm: An open-weights llm for research on code generation with world models. *arXiv preprint arXiv:2510.02387*.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385.

Wenlong Deng, Yushu Li, Boying Gong, Yi Ren, Christos Thrampoulidis, and Xiaoxiao Li. 2025. On grpo collapse in search-r1: The lazy likelihood-displacement death spiral. *arXiv preprint arXiv:2512.04220*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114.

Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, and 1 others. 2025. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys*, 58(3):1–38.

Shangheng Du, Jiabao Zhao, Jinxin Shi, Zhentao Xie, Xin Jiang, Yanhong Bai, and Liang He. 2026. A survey on the optimization of large language model-based agents. *ACM Computing Surveys*, 58(9):1–37.

Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, and 1 others. 2025a. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv preprint arXiv:2508.07407*.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025b. WebEvolver: Enhancing web agent self-improvement with co-evolving world model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8970–8986, Suzhou, China. Association for Computational Linguistics.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025. Group-in-group policy optimization for LLM agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Saumya Gandhi, Ritu Gala, Vijay Viswanathan, Tongshuang Wu, and Graham Neubig. 2024. Better synthetic data by retrieving and transforming existing datasets. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6453–6466, Bangkok, Thailand. Association for Computational Linguistics.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, and 1 others. 2025a. A survey of self-evolving agents: On path to artificial super intelligence. *arXiv preprint arXiv:2507.21046*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025b. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *The Innovation*.

Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, Huan Sun, and Yu Su. 2025. Is your LLM secretly a world model of the internet? model-based planning for web agents. *Transactions on Machine Learning Research*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jiacheng Guo, Ling Yang, Peter Chen, Qixin Xiao, Yinjie Wang, Xinzhe Juan, Jiahao Qiu, Ke Shen, and Mengdi Wang. 2025b. Genenv: Difficulty-aligned co-evolution between llm agents and environment simulators. *Preprint*, arXiv:2512.19682.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8048–8057. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122*.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Hongming Zhang, Tianqing Fang, Zhenzhong Lan, and Dong Yu. 2025. OpenWebVoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27545–27564, Vienna, Austria. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Haoyang Hong, Jiajun Yin, Yuan Wang, Jingnan Liu, Zhe Chen, Ailing Yu, Ji Li, Zhiling Ye, Hansong Xiao, Yefei Chen, and 1 others. 2025. Multi-agent deep research: Training multi-agent systems with m-grpo. *arXiv preprint arXiv:2511.13288*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. 2025. TreeRL: LLM reinforcement learning with on-policy tree search. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12355–12369, Vienna, Austria. Association for Computational Linguistics.

Minda Hu, Tianqing Fang, Jianshu Zhang, Jun-Yu Ma, Zhisong Zhang, Jingyan Zhou, Hongming Zhang, Haitao Mi, Dong Yu, and Irwin King. 2025a. WebCoT: Enhancing web agent reasoning by reconstructing chain-of-thought in reflection, branching, and rollback. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5155–5173, Suzhou, China. Association for Computational Linguistics.

Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, and 1 others. 2025b. Os agents: A survey on mllm-based agents for general computing devices use. *arXiv preprint arXiv:2508.04482*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*.

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, and 1 others. 2025. Verltool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*.

Minki Kang, Jongwon Jeong, Seanie Lee, Jaewoong Cho, and Sung Ju Hwang. 2025. Distilling llm agent into small models with retrieval and code tools. *Advances in Neural Information Processing Systems*, 38:106501–106538.

Imene Kerboua, Sahar Omidi Shayegan, Megh Thakkar, Xing Han Lù, Léo Boisvert, Massimo Caccia, Jérémy Espinas, Alexandre Aussem, Véronique Eglin, and Alexandre Lacoste. 2025. Focusagent: Simple yet effective ways of trimming the large context of web agents. *arXiv preprint arXiv:2510.03204*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Jing Yu Koh, Stephen Marcus McAleer, Daniel Fried, and Ruslan Salakhutdinov. 2025. Tree search for language model agents. *Transactions on Machine Learning Research*.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and 1 others. 2024. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5295–5306.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, and 1 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*.

Dongjun Lee, Juyong Lee, Kyuyoung Kim, Jihoon Tack, Jinwoo Shin, Yee Whye Teh, and Kimin Lee. 2025. Learning to contextualize web pages for enhanced decision making by LLM agents. In *The Thirteenth International Conference on Learning Representations*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Ren Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 26874–26901. PMLR.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Chenliang Li, Adel Elmahdy, Alex Boyd, Zhongruo Wang, Alfredo Garcia, Parminder Bhatia, Taha Kass-Hout, Cao Xiao, and Mingyi Hong. 2025a. Stppo: Stabilized off-policy proximal policy optimization for multi-turn agents training. *arXiv preprint arXiv:2511.20718*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, and 1 others. 2025b. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*.

Shiyu Li, Yang Tang, Yifan Wang, Peiming Li, and Xi Chen. 2025c. Reseek: A self-correcting framework for search agents with instructive rewards. *arXiv preprint arXiv:2510.00568*.

Wei Li, William E Bishop, Alice Li, Christopher Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024. On the effects of data scale on ui control agents. *Advances in Neural Information Processing Systems*, 37:92130–92154.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Zongyu Lin, Yao Tang, Xingcheng Yao, Da Yin, Ziniu Hu, Yizhou Sun, and Kai-Wei Chang. 2025a. QLASS: Boosting language agent inference via q-guided stepwise search. In *Forty-second International Conference on Machine Learning*.

Zongyu Lin, Yao Tang, Xingcheng Yao, Da Yin, Ziniu Hu, Yizhou Sun, and Kai-Wei Chang. 2025b. QLASS: Boosting language agent inference via q-guided stepwise search. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 37942–37958. PMLR.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, and 1 others. 2025b. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*.

Junwei Liu, Kaixin Wang, Yixuan Chen, Xin Peng, Zhenpeng Chen, Lingming Zhang, and Yiling Lou. 2024. Large language model-based agents for software engineering: A survey. *ACM Transactions on Software Engineering and Methodology*.

Keliang Liu, Dingkang Yang, Ziyun Qian, Weijie Yin, Yuchi Wang, Hongsheng Li, Jun Liu, Peng Zhai, Yang Liu, and Lihua Zhang. 2025c. Reinforcement learning meets large language models: A survey of advancements and applications across the llm lifecycle. *arXiv preprint arXiv:2509.16679*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35.

Xiaoqian Liu, Ke Wang, Yuchuan Wu, Fei Huang, Yongbin Li, Junge Zhang, and Jianbin Jiao. 2025d. Agentic reinforcement learning with implicit step rewards. *arXiv preprint arXiv:2509.19199*.

Han Lu, Zichen Liu, Shaopan Xiong, Yancheng He, Wei Gao, Yanan Wu, Weixun Wang, Jiashun Liu, Yang Li, Haizhou Zhao, and 1 others. 2025a. Part ii: Roll flash–accelerating rlvr and agentic training with asynchrony. *arXiv preprint arXiv:2510.11345*.

Miao Lu, Weiwei Sun, Weihua Du, Zhan Ling, Xuesong Yao, Kang Liu, and Jiecao Chen. 2025b. Scaling llm multi-turn rl with end-to-end summarization-based context management. *arXiv preprint arXiv:2510.06727*.

Rui Lu, Zhenyu Hou, Zihan Wang, Hanchen Zhang, Xiao Liu, Yujiang Li, Shi Feng, Jie Tang, and Yuxiao Dong. 2025c. Deepdive: Advancing deep search agents with knowledge graphs and multi-turn rl. *arXiv preprint arXiv:2509.10446*.

Xing Han Lù, Gaurav Kamath, Marius Mosbach, and Siva Reddy. 2025. Build the web for agents, not agents for the web. *arXiv preprint arXiv:2506.10953*.

Xing Han Lu, Zdeněk Kasner, and Siva Reddy. 2024a. WebLINX: Real-world website navigation with multi-turn dialogue. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 33007–33056. PMLR.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. 2024b. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, and 1 others. 2025a. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*.

Kun Luo, Hongjin Qian, Zheng Liu, Ziyi Xia, Shitao Xiao, Siqi Bao, Jun Zhao, and Kang Liu. 2025b. Infoflow: Reinforcing search agent via reward density optimization. *arXiv preprint arXiv:2510.26575*.

Yuanjie Lyu, Chengyu Wang, Jun Huang, and Tong Xu. 2025. From correction to mastery: Reinforced distillation of large language model agents. *arXiv preprint arXiv:2509.14257*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, and 1 others. 2025. A survey of context engineering for large language models. *arXiv preprint arXiv:2507.13334*.

Grégoire Mialon, Roberto Dessi, Maria Lomeli, Christoforos Nalmpantis, Ramakanth Pasunuru, Roberta Raileanu, Baptiste Roziere, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented language models: a survey. *Transactions on Machine Learning Research*. Survey Certification.

Sumeet Ramesh Motwani, Chandler Smith, Rocktim Jyoti Das, Rafael Rafailov, Philip Torr, Ivan Laptev, Fabio Pizzati, Ronald Clark, and Christian Schroeder de Witt. 2025. MALT: Improving reasoning with multi-agent LLM training. In *Second Conference on Language Modeling*.

Shikhar Murty, Christopher D Manning, Peter Shaw, Mandar Joshi, and Kenton Lee. 2024a. Bagel: Bootstrapping agents by guiding exploration with language. In *International Conference on Machine Learning*, pages 36894–36910. PMLR.

Shikhar Murty, Hao Zhu, Dzmitry Bahdanau, and Christopher D Manning. 2024b. Nnetnav: Unsupervised learning of browser agents through environment interaction in the wild. *arXiv preprint arXiv:2410.02907*.

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, Xintong Li, Jing Shi, Hongjie Chen, Viet Dac Lai, Zhouhang Xie, Sungchul Kim, Ruiyi Zhang, Tong Yu, Mehrab Tanjim, and 11 others. 2025a. GUI agents: A survey. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 22522–22538, Vienna, Austria. Association for Computational Linguistics.

Dang Nguyen, Viet Dac Lai, Seunghyun Yoon, Ryan A. Rossi, Handong Zhao, Ruiyi Zhang, Puneet Mathur, Nedim Lipka, Yu Wang, Trung Bui, Franck Dernoncourt, and Tianyi Zhou. 2025b. Dynasaur: Large language agents beyond predefined actions. In *Second Conference on Language Modeling*.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, and 1 others. 2025. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6140–6150.

Tianyue Ou, Frank F Xu, Aman Madaan, Jiarui Liu, Robert Lo, Abishek Sridhar, Sudipta Sengupta, Dan Roth, Graham Neubig, and Shuyan Zhou. 2024. Synatra: Turning indirect knowledge into direct demonstrations for digital agents at scale. *Advances in Neural Information Processing Systems*, 37:91618–91652.

Vardaan Pahuja, Yadong Lu, Corby Rosset, Boyu Gou, Arindam Mitra, Spencer Whitehead, Yu Su, and Ahmed Hassan Awadallah. 2025. Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6300–6323, Vienna, Austria. Association for Computational Linguistics.

Chanwoo Park, Seungju Han, Xingzhi Guo, Asuman E. Ozdaglar, Kaiqing Zhang, and Joo-Kyung Kim. 2025. MAPoRL: Multi-agent post-co-training for collaborative large language models with reinforcement learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30215–30248, Vienna, Austria. Association for Computational Linguistics.

Ajay Patel, Markus Hofmarcher, Claudiu Leoveanu-Condrei, Marius-Constantin Dinu, Chris Callison-Burch, and Sepp Hochreiter. 2024. Large language models can self-improve at web agent tasks. *arXiv preprint arXiv:2405.20309*.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2024. Gorilla: Large language model connected with massive APIs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. 2025. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *The Thirteenth International Conference on Learning Representations*.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, Bangkok, Thailand. Association for Computational Linguistics.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru WANG, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. ToolRL: Reward is all tool learning needs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, and 1 others. 2024. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025a. Tool learning with large language models: A survey. *Frontiers of Computer Science*, 19(8):198343.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Ruslan Salakhutdinov, Chelsea Finn, and Aviral Kumar. 2025b. Rlad: Training llms to discover abstractions for solving reasoning problems. *arXiv preprint arXiv:2510.02263*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ram Ramrakhya, Andrew Szot, Omar Attia, Yuhao Yang, Anh Nguyen, Bogdan Mazoure, Zhe Gan, Harsh Agrawal, and Alexander Toshev. 2025. Scaling synthetic task generation for agents via exploration. *arXiv preprint arXiv:2509.25047*.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems*, 36:59708–59728.

Jérémy Scheurer, Jon Ander Campos, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2022. Training language models with language feedback. *arXiv preprint arXiv:2204.14146*.

Leyang Shen, Yang Zhang, Chun Kai Ling, Xiaoyan Zhao, and Tat-Seng Chua. 2025. Carl: Critical action focused reinforcement learning for multi-step agent. *arXiv preprint arXiv:2512.04949*.

Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, and 1 others. 2025a. Taskcraft: Automated generation of agentic tasks. *arXiv preprint arXiv:2506.10055*.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2024. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*.

Yaorui Shi, Yuxin Chen, Siyuan Wang, Sihang Li, Hengxing Cai, Qi Gu, Xiang Wang, and An Zhang. 2025b. Look back to reason forward: Revisitable memory for long-context llm agents. *arXiv preprint arXiv:2509.23040*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. {ALFW}orld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025a. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7584–7600, Bangkok, Thailand. Association for Computational Linguistics.

Yueqi Song, Ketan Ramaneti, Zaid Sheikh, Ziru Chen, Boyu Gou, Tianbao Xie, Yiheng Xu, Danyang Zhang, Apurva Gandhi, Fan Yang, and 1 others. 2025b. Agent data protocol: Unifying datasets for diverse, effective fine-tuning of llm agents. *arXiv preprint arXiv:2510.24702*.

Yueqi Song, Frank F. Xu, Shuyan Zhou, and Graham Neubig. 2025c. Beyond browsing: API-based web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11066–11085, Vienna, Austria. Association for Computational Linguistics.

Hongjin Su, Ruoxi Sun, Jinsung Yoon, Pengcheng Yin, Tao Yu, and Sercan O Arik. 2025a. Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments. In *The Thirteenth International Conference on Learning Representations*.

Liangcai Su, Zhen Zhang, Guangyu Li, Zhuo Chen, Chenxi Wang, Maojia Song, Xinyu Wang, Kuan Li, Jialong Wu, Xuanzhong Chen, and 1 others. 2025b. Scaling agents via continual pre-training. *arXiv preprint arXiv:2509.13310*.

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. 2025a. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, Ben Kao, Guohao Li, Junxian He, Yu Qiao, and Zhiyong Wu. 2025b. OS-genesis: Automating GUI agent trajectory construction via reverse task synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5555–5579, Vienna, Austria. Association for Computational Linguistics.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. 2025c. Scaling long-horizon llm agent via context-folding. *arXiv preprint arXiv:2510.11967*.

Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA.

Sijun Tan, Michael Luo, Colin Cai, Tarun Venkat, Kyle Montgomery, Aaron Hao, Tianhao Wu, Arnav Balyan, Manan Roongta, Chenguang Wang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025a. rllm: A framework for post-training language agents. Notion Blog.

Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T Liu, Philipp Koehn, and Lu Lu. 2025b. Process-supervised reinforcement learning for interactive multimodal tool-use agents. *arXiv preprint arXiv:2509.14480*.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 930–957, Miami, Florida, USA. Association for Computational Linguistics.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, and 1 others. 2025a. A survey on (m)llm-based gui agents. *arXiv preprint arXiv:2504.13865*.

Qiaoyu Tang, Hao Xiang, Le Yu, Bowen Yu, Yaojie Lu, Xianpei Han, Le Sun, WenJuan Zhang, Pengbo Wang, Shixuan Liu, and 1 others. 2025b. Beyond turn limits: Training deep search agents with dynamic context window. *arXiv preprint arXiv:2510.08276*.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, and 1 others. 2025c. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, and 1 others. 2025. Webshaper: Agentically data synthesizing via information-seeking formalization. *arXiv preprint arXiv:2507.15061*.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Guiyao Tie, Zeli Zhao, Dingjie Song, Fuyang Wei, Rong Zhou, Yurou Dai, Wen Yin, Zhejian Yang, Jiangyue Yan, Yao Su, and 1 others. 2025. Large language models post-training: Surveying techniques from alignment to reasoning. *arXiv preprint arXiv:2503.06072*.

Brandon Trabucco, Gunnar Sigurdsson, Robinson Piramuthu, and Ruslan Salakhutdinov. 2025. Insta: Towards internet-scale training for agents. *arXiv preprint arXiv:2502.06776*.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Dheeraj Vattikonda, Santhoshi Ravichandran, Emiliano Penaloza, Hadi Nekoei, Megh Thakkar, Thibault Le Sellier de Chezelles, Nicolas Gontier, Miguel Muñoz-Mármol, Sahar Omidi Shayegan, Stefania Raimondo, and 1 others. 2025. How to train your llm web agent: A statistical diagnosis. *arXiv preprint arXiv:2507.04103*.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. 2025a. Information gain-based policy optimization: A simple and effective approach for multi-turn llm agents. *arXiv preprint arXiv:2510.14967*.

Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, and 1 others. 2025b. Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning. *arXiv preprint arXiv:2509.02544*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2024b. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383.

Rui Wang, Ce Zhang, Jun-Yu Ma, Jianshu Zhang, Hongru Wang, Yi Chen, Boyang Xue, Tianqing Fang, Zhisong Zhang, Hongming Zhang, and 1 others. 2025c. Explore to evolve: Scaling evolved aggregation logic via proactive online exploration for deep research agents. *arXiv preprint arXiv:2510.14438*.

Ruiyi Wang and Prithviraj Ammanabrolu. 2025. A practitioner's guide to multi-turn agentic reinforcement learning. *arXiv preprint arXiv:2510.01132*.

Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. 2024c. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400*.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024d. Executable code actions elicit better LLM agents. In *Forty-first International Conference on Machine Learning*.

Yanlin Wang, Wanjun Zhong, Yanxian Huang, Ensheng Shi, Min Yang, Jiachi Chen, Hui Li, Yuchi Ma, Qianxiang Wang, and Zibin Zheng. 2025d. Agents in software engineering: Survey, landscape, and vision. *Automated Software Engineering*, 32(2):1–36.

Yiming Wang, Da Yin, Yuedong Cui, Ruichen Zheng, Zhiqian Li, Zongyu Lin, Di Wu, Xueqing Wu, Chenchen Ye, Yu Zhou, and 1 others. 2025e. Llms as scalable, general-purpose simulators for evolving digital agent training. *arXiv preprint arXiv:2510.14969*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, and 1 others. 2025f. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.

Ziliang Wang, Kang An, Xuhui Zheng, Faqiang Qian, Weikun Zhang, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. 2025g. Erase to improve: Erasable reinforcement learning for search-augmented llms. *arXiv preprint arXiv:2510.00861*.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, LINGMING ZHANG, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida Wang. 2025. SWE-RL: Advancing LLM reasoning via reinforcement learning on open software evolution. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Feijie Wu, Weiwu Zhu, Yuxiang Zhang, Soumya Chatterjee, Jiarong Zhu, Fan Mo, Rodin Luo, and Jing Gao. 2025a. Portool: Tool-use llm training with rewarded tree. *arXiv preprint arXiv:2510.26020*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhenglin Wang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Xiangru Tang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025b. Webdancer: Towards autonomous information seeking agency. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. 2025c. Hiprag: Hierarchical process rewards for efficient agentic retrieval augmented generation. *arXiv preprint arXiv:2510.07794*.

Peilin Wu, Mian Zhang, Xinlu Zhang, Xinya Du, and Zhiyu Chen. 2025d. Search wisely: Mitigating sub-optimal agentic searches by reducing uncertainty. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19734–19745, Suzhou, China. Association for Computational Linguistics.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, and 1 others. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Xinmiao Yu, Dingchu Zhang, Yong Jiang, and 1 others. 2025e. Resum: Unlocking long-horizon search intelligence via context summarization. *arXiv preprint arXiv:2509.13313*.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. 2025a. A survey of llm-based deep search agents: Paradigm, optimization, evaluation, and challenges. *arXiv preprint arXiv:2508.05668*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025b. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Yajie Yang, Wenxiang Chen, Zhihao Zhang, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, and 1 others. 2025c. Agentprm: Process reward models for llm agents via step-wise promise and progress. *arXiv preprint arXiv:2511.08325*.

Jingxu Xie, Dylan Xu, Xuandong Zhao, and Dawn Song. 2025a. Agentsynth: Scalable task generation for generalist computer-use agents. *arXiv preprint arXiv:2506.14205*.

Junlin Xie, Zhihong Chen, Ruifei Zhang, and Guanbin Li. 2025b. Large multimodal agents: a survey. *Visual Intelligence*, 3(1):24.

Weimin Xiong, Yifan Song, Qingxiu Dong, Bingchan Zhao, Feifan Song, XWang, and Sujian Li. 2025. MPO: Boosting LLM agents with meta plan optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 3914–3935, Suzhou, China. Association for Computational Linguistics.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. 2024. Watch every step! LLM agent learning via iterative step-level process refinement. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1556–1572, Miami, Florida, USA. Association for Computational Linguistics.

Weiwen Xu, Deng Cai, Zhisong Zhang, Wai Lam, and Shuming Shi. 2024. Reasons to reject? aligning language models with judgments. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12288–12304, Bangkok, Thailand. Association for Computational Linguistics.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. 2025. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. In *The Thirteenth International Conference on Learning Representations*.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Kristian Kersting, Jeff Z Pan, Hinrich Schütze, and 1 others. 2025. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2025a. Agentoccam: A simple yet strong baseline for LLM-based web agents. In *The Thirteenth International Conference on Learning Representations*.

Ruihan Yang, Fanghua Ye, Jian Li, Siyu Yuan, Yikai Zhang, Zhaopeng Tu, Xiaolong Li, and Deqing Yang. 2025b. The lighthouse of language: Enhancing LLM agents via critique-guided improvement. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zonghan Yang, Peng Li, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. 2024. React meets actre: Autonomous annotation of agent trajectories for contrastive self-training. In *First Conference on Language Modeling*.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Liwen Zhang, Zile Qiao, Xinyu Wang, and 1 others. 2025. Agentfold: Long-horizon web agents with proactive context management. *arXiv preprint arXiv:2510.24699*.

Asaf Yehudai, Lilach Eden, Alan Li, Guy Uziel, Yilun Zhao, Roy Bar-Haim, Arman Cohan, and Michal Shmueli-Scheuer. 2025. Survey on evaluation of llm-based agents. *arXiv preprint arXiv:2503.16416*.

Howard Yen, Ashwin Paranjape, Mengzhou Xia, Thejas Venkatesh, Jack Hessel, Danqi Chen, and Yuhao Zhang. 2025. Lost in the maze: Overcoming context limitations in long-horizon agentic search. *arXiv preprint arXiv:2510.18939*.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. 2024a. Agent lumos: Unified and modular training for open-source language agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12380–12403, Bangkok, Thailand. Association for Computational Linguistics.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2024b. A survey on multimodal large language models. *National Science Review*, 11(12):nwae403.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. 2026. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. *arXiv preprint arXiv:2601.01885*.

Yuanqing Yu, Zhefan Wang, Weizhi Ma, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. 2025a. Steptool: Enhancing multi-step tool usage in llms via step-grained reinforcement learning. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 3952–3962.

Zhaochen Yu, Ling Yang, Jiaru Zou, Shuicheng Yan, and Mengdi Wang. 2025b. Demystifying reinforcement learning in agentic reasoning. *arXiv preprint arXiv:2510.11701*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. 2024. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3053–3077, Bangkok, Thailand. Association for Computational Linguistics.

Yunpeng Zhai, Shuchang Tao, Cheng Chen, Anni Zou, Ziqian Chen, Qingxu Fu, Shinji Mai, Li Yu, Jiaji Deng, Zouying Cao, and 1 others. 2025. Agentevolver: Towards efficient self-evolving agent system. *arXiv preprint arXiv:2511.10395*.

Chaoyun Zhang, Shilin He, Liqun Li, Si Qin, Yu Kang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. 2025a. Api agents vs. gui agents: Divergence and convergence. *arXiv preprint arXiv:2503.11069*.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2025b. Large language model-brained GUI agents: A survey. *Transactions on Machine Learning Research*.

Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2025c. UFO: A UI-focused agent for windows OS interaction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 597–622, Albuquerque, New Mexico. Association for Computational Linguistics.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, and 1 others. 2025d. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. 2025e. Agentracer: Who is inducing failure in the llm agentic systems? *arXiv preprint arXiv:2509.03312*.

Jianguo Zhang, Tian Lan, Rithesh Murthy, Zhiwei Liu, Weiran Yao, Ming Zhu, Juntao Tan, Thai Hoang, Zuxin Liu, Liangwei Yang, and 1 others. 2024. Agentohana: Design unified data and training pipeline for effective agent learning. *arXiv preprint arXiv:2402.15506*.

Jiayuan Zhang, Kaiquan Chen, Zhihao Lu, Enshen Zhou, Qian Yu, and Jing Zhang. 2025f. Prune4web: Dom tree pruning programming for web agent. *arXiv preprint arXiv:2511.21398*.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, and 1 others. 2025g. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. 2025h. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *Forty-second International Conference on Machine Learning*.

Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. 2025i. Deep research: A survey of autonomous research agents. *arXiv preprint arXiv:2508.12752*.

Yaocheng Zhang, Haohuan Huang, Zijun Song, Yuanheng Zhu, Qichao Zhang, Zijie Zhao, and Dongbin Zhao. 2025j. Criticsearch: Fine-grained credit assignment for search agents via a retrospective critic. *arXiv preprint arXiv:2511.12159*.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2025k. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yida Zhao, Kuan Li, Xixi Wu, Liwen Zhang, Dingchu Zhang, Baixuan Li, Maojia Song, Zhuo Chen, Chenxi Wang, Xinyu Wang, and 1 others. 2025. Repurposing synthetic data for fine-grained search agent supervision. *arXiv preprint arXiv:2510.24694*.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024a. Gpt-4v (ision) is a generalist web agent, if grounded. In *International Conference on Machine Learning*, pages 61349–61385. PMLR.

Congming Zheng, Jiachen Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, and 1 others. 2025a. A survey of process reward models: From outcome signals to process supervisions for large language models. *arXiv preprint arXiv:2510.08049*.

Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. 2025b. Lifelong learning of large language model based agents: A roadmap. *arXiv preprint arXiv:2501.07278*.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2024b. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*.

Xuhui Zheng, Kang An, Ziliang Wang, Yuhang Wang, and Yichao Wu. 2025c. StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21816–21841, Suzhou, China. Association for Computational Linguistics.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024a. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 62138–62160.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024b. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*.

Yifei Zhou, Qianlan Yang, Kaixiang Lin, Min Bai, Xiong Zhou, Yu-Xiong Wang, Sergey Levine, and Li Erran Li. 2025. Proposer-agent-evaluator (PAE): Autonomous skill discovery for foundation model internet agents. In *Forty-second International Conference on Machine Learning*.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. 2024c. ArCHer: Training language model agents via hierarchical multi-turn RL. In *Forty-first International Conference on Machine Learning*.

Chenyang Zhu, Spencer Hong, Jingyu Wu, Kushal Chawla, Charlotte Tang, Youbing Yin, Nathan Wolfe, Erin Babinsky, and Daben Liu. 2025. Raffles: Reasoning-based attribution of faults for llm systems. *arXiv preprint arXiv:2509.06822*.

A Paper Collection Methodology

Our survey method consists of a combination of broad trend estimation, targeted paper retrieval, automatic-assisted screening, manual verification, and snowballing. We cover two primary goals: estimating the publication trend of LLM-based agent training papers, and constructing the final corpus of papers discussed in this survey.

Trend estimation. To characterize the growth of the research direction, we first collect arXiv papers from the computer science categories using a broad keyword search with the term "agent". This broad query is intended to provide a high-recall estimate of the overall growth of agent-related papers rather than to define the final survey corpus. We then apply a two-stage filtering procedure. First, we retrieve candidate papers using the keyword search. Second, we use QWEN3-8B as an automatic screening assistant to judge from the title and abstract whether each candidate paper is related to LLM-based agent training (prompt can be found at the end of this sub-section). The resulting quarterly counts are shown in Figure 2, which illustrates the rapid growth of research on LLM-based agents and agent training.

Survey corpus construction. The final survey corpus is constructed through a broader and more targeted process. In addition to arXiv, we additionally search major publication venues and indexing sources relevant to NLP. We use a larger set of keywords related to LLM-based agents and training, including combinations of words such as "agent", "training", "learning" and so on. We further supplement the search results by snowballing from references of closely related surveys and representative papers.

Inclusion and exclusion criteria. We include a paper if it satisfies the following criteria: (1) it studies agents based on LLMs; (2) the agent interacts with an external environment; and (3) the paper involves training-related methodology. We exclude papers that focus only on prompt engineering without policy updates or general LLM training without environment interaction.

Screening and validation. The screening process proceeds in three stages. First, we remove clearly irrelevant papers based on title and abstract. Second, we use LLM-assisted filtering to improve recall and reduce manual workload. The automatic screening result is used only as an auxiliary signal rather than a final inclusion decision. Third, the authors manually verify all papers included in the final survey corpus by further reading. For borderline papers, we check whether the paper contributes to at least one stage of the agent training pipeline considered in this survey: environment setup, data and trajectory construction, learning-signal design, or optimization objective and training scheme.

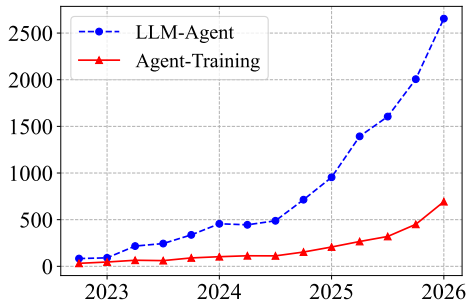

Figure 2: Trends of LLM-based agent research on arXiv. The data is aggregated per season (three months), illustrating the number of publications on general LLM-based agents and on dedicated agent training methods.

Scope of the collected papers. Consistent with the scope of this survey, our primary focus is on methods that update the parameters of the main LLM-based agent policy. We also include papers on data construction, environment design, and feedback generation when they directly support such policy training. Related topics such as prompting-based agents, external memory, planning modules, evaluation and benchmark design are discussed when they influence agent policy training, but they are not the main basis for inclusion.

PROMPT FOR PAPER COLLECTING

```
Role: You are an expert research assistant
with a specialization in Large Language
Models and AI Agents. Your task is to
carefully read a paper's title and abstract
and classify it according to a precise set
of criteria.

Task: You will be given the title and
abstract of a research paper. Based only
on this information, you must:
  • Determine if the paper is about LLM
    Agents.
  • Determine if the paper involves Agent
    Training.
  • Extract a list of the paper's main
    keywords.
  • Respond only with a single,
    valid JSON object containing your
    classifications.
```

```
Definitions:
  1. "is_agent": (bool) Set this to true
     if the paper describes an LLM-based
     system that exhibits autonomous
     behavior. Mark this as false if no
     LLM is utilized.
  2. "is_training": (bool) This field is
     dependent on the first. If "is_agent"
     is false, this must also be false. If
     "is_agent" is true, set this to true
     only if the paper discusses modifying
     the LLM's model weights to improve its
     agentic capabilities.
     • Includes: Fine-tuning
       (full-parameter), Instruction
       tuning (on agent-specific data),
       Reinforcement Learning (e.g., DPO,
       PPO, GRPO), Parameter-Efficient
       Fine-Tuning (PEFT, e.g., LoRA,
       QLoRA), Adapter training, or
       training a new agent-specific model
       from scratch.
     • Excludes (NOT training): Prompt
       Engineering (e.g., ReAct, Chain
       of Thought), In-Context Learning
       (Zero-shot, Few-shot), using a
       static pre-trained LLM as a "brain"
       without updating its weights, or
       building a system around a black-box
       LLM.
  3. "keywords": (list[str]) Extract a list
     of 3-5 of the most important and
     specific keywords or phrases from
     the abstract (e.g., "fine-tuning",
     "tool use", "autonomous agent", "ReAct
     framework").

Input:
Title: {title}
Abstract: {abstract}

Output Format (JSON Only): Your response
must be only the JSON object, with no
introductory text or explanation.
{
    "is_agent": bool,
    "is_training": bool,
    "keywords": ["keyword1", "keyword2",
"keyword3"]
}
```

B Other Aspects

Training Target. While our focus is on the training of the main policy model, which is the component responsible for deciding the next action based on the current input, an agent system often contains **other trainable components** that also require dedicated training. These auxiliary components can be trained in similar ways to achieve specialized functionalities, such as managing memory (Zhang et al., 2025k; Yan et al., 2025; Shi et al., 2025b; Yu et al., 2026), performing explicit error correction (Wang et al., 2025g), or generating improved plans (Qu et al., 2025b; Xiong et al., 2025). For systems structured with multiple components, the training signals for a specific component can often be obtained by running the full system while keeping other components frozen. The complexity may be further amplified in **multi-agent systems** (Guo et al., 2024; Tran et al., 2025), which have become widely adopted, and whose effective training requires careful consideration (Albrecht et al., 2024; Park et al., 2025; Motwani et al., 2025; Hong et al., 2025).

Prompting. While this work has primarily focused on traditional model learning approaches that involve tuning model parameters, the remarkable advancements in LLMs necessitate discussing methods that leverage their inherent capabilities. Specifically, the strong in-context learning and instruction-following abilities of modern LLMs enable significant improvements in agent systems through prompting (Liu et al., 2023). The key advantage of **prompting-based** approaches is their flexibility and light-weight nature. They can be easily adopted to accumulate experience (Tang et al., 2025c), perform reflection (Shinn et al., 2023; Madaan et al., 2023) and incorporate external knowledge (Mialon et al., 2023).

Environment Modeling. While the primary goal of agent training is to learn a good policy $\pi(a_t|s_t)$ that decides the next action, an agent system may also benefit from training a model that predicts the transition of the environment $p(s_{t+1}|s_t, a_t)$. This component, often referred to as a world model (Ha and Schmidhuber, 2018), predicts the next state s_{t+1} given the current state s_t and the action taken a_t. Such **environment modeling** can be directly used in the agent's planning process, enabling decisions to be made with a better internal simulation and awareness of potential future outcomes (Chae et al., 2025a; Gu et al., 2025). Furthermore, it can also enhance the training process through better data synthesis (Fang et al., 2025b) and environment synthesis (Guo et al., 2025b; Liu et al., 2025a), or by serving as the target for specialized intermediate pre-training stages (Copet et al., 2025).

C About AI Assistants

AI is only used for linguistic polishing; the conceptual framework, literature synthesis, and organization are entirely the original work of the authors.

D Background

D.1 LLM-Based Agents

An LLM-based agent is an AI system designed to interact with the environment to achieve target goals. These agents leverage the inherent understanding and reasoning capabilities of LLMs and interact with the environment in an iterative way. At each time step t, the agent receives a representation of the environment's current state $s_t \in \mathcal{S}$ and determines an action $a_t \in \mathcal{A}$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. The action decision is made by the underlying LLM, which processes the environmental observations and the interaction history as input, and generates the action decision as output. The action will be executed, leading the environment to a new state s_{t+1}, where the agent will make further decisions. The interaction will continue until a termination state s_T, resulting in a sequence of states and actions, formally known as a trajectory: $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$.

D.2 Agent Training

Although LLMs have remarkable instruction-following capabilities, allowing them to be directly prompted for action decisions without specific parameter updates, the effectiveness of this approach is constrained, especially when the LLM is unfamiliar with the state and action representations of the target environment. To enable more stable and improved results, dedicated agent training is usually essential. While the overall agent training process shares similarities with the ordinary LLM training, the multi-step environmental interactions required by agent tasks introduce specific considerations.

E Summary and Guidance

We conclude this survey by synthesizing the related work across the key stages of the agent training pipeline. Rather than enumerating individual methods, this section summarizes recurring design choices, trade-offs, and applicability conditions in environment setup, data and trajectory construction, learning-signal design, and optimization schemes. The following tables are intended to provide a compact reference for understanding when different agent-training strategies are suitable, what assumptions they rely on, and what limitations they introduce.

Environment	Observation space	Action space	Feedback	Interaction cost
QA/Search	Retrieved docs	Search	Partially verifiable	Low/Medium
Tool/API	Tool outputs	Function/API calls	Often verifiable	Low/Medium
Code/SWE	Compiler/Runtime outputs	Code edits/Commands	Verifiable	Medium
Web	Web pages	Browser actions	Maybe verifiable, often delayed	Medium/High
GUI	Screenshots/UI elements	UI actions	Often sparse and task-level	Medium/High
Embodied	Visual/Sensor states	Embodied actions	Task dependent	High

Table 1: Environment-level differences that shape LLM-based agent training. Environment design determines the observation and action spaces, the availability of feedback and the cost of interaction.

Stage	Strategy	Grounding	Cost	Main trade-off
Query	Direct synthesis	Low	Low	Highly scalable and controllable, but may produce simple or weakly grounded tasks.
Query	Context-aware synthesis	Medium	Low/Medium	Improves environmental relevance, but may still miss interaction-level complexity.
Query	Iterative extension	Medium	Medium	Increases difficulty and compositionality, but may reduce naturalness.
Query	Exploratory construction	High	High	Strongly grounded in environment dynamics, but costly and harder to control.
Trajectory	Trajectory conversion	Medium	Low/Medium	Reuses existing resources, but depends on compatible source data and formats.
Trajectory	Teacher-model sampling	High	Medium/High	Produces strong demonstrations, but can be expensive and teacher-biased.
Trajectory	Self-sampling	High	Medium/High	Matches learner distribution, but early trajectories can be low quality.
Trajectory	Search-based sampling	High	High	Improves exploration and path quality, but increases interaction and computation cost.

Table 2: Trade-offs in query and trajectory construction for agent training. Query construction controls task goals, while trajectory construction supplies the interaction process used for policy learning.

Signal Source	Verifiability	Sparsity	Noise risk	Suitable settings
Expert annotation	Medium	Medium	Medium	Open-ended or subjective agent behavior
Success reward	High	High	Low	Goal-directed tasks with clear success criteria
Execution feedback	High	Medium/High	Low	Code, tool-use, API, database, and web tasks
LLM-as-judge	Medium	Medium	High	Scalable evaluation when human labels are expensive
Self-reward	Low	Medium	High	Bootstrapping and iterative self-improvement

Table 3: Trade-offs among learning signals sources for agent training. Agent feedback differs from standard LLM feedback because it is often environment-dependent, delayed, and difficult to assign to individual actions.