Intelligent Agents with LLM-based Process Automation

Yanchu Guan*
yanchu.gyc@antgroup.com
Ant Group
Hangzhou, China

Dong Wang*
yishan.wd@antgroup.com
Ant Group
Hangzhou, China

Zhixuan Chu*,†
zhixuanchu@zju.edu.cn
Zhejiang University
Hangzhou, China

Shiyu Wang*
weiming.wsy@antgroup.com
Ant Group
Hangzhou, China

Feiyue Ni
nifeiyue@ruc.edu.cn
Renmin University of China
Beijing, China

Ruihua Song
rsong@ruc.edu.cn
Renmin University of China
Beijing, China

Chenyi Zhuang
chenyi.zcy@antgroup.com
Ant Group
Hangzhou, China

ABSTRACT

While intelligent virtual assistants like Siri, Alexa, and Google Assistant have become ubiquitous in modern life, they still face limitations in their ability to follow multi-step instructions and accomplish complex goals articulated in natural language. However, recent breakthroughs in large language models (LLMs) show promise for overcoming existing barriers by enhancing natural language processing and reasoning capabilities. Though promising, applying LLMs to create more advanced virtual assistants still faces challenges like ensuring robust performance and handling variability in real-world user commands. This paper proposes a novel LLM-based virtual assistant that can automatically perform multi-step operations within mobile apps based on high-level user requests. The system represents an advance in assistants by providing an end-to-end solution for parsing instructions, reasoning about goals, and executing actions. LLM-based Process Automation (LLMPA) has modules for decomposing instructions, generating descriptions, detecting interface elements, predicting next actions, and error checking. Experiments demonstrate the system completing complex mobile operation tasks in Alipay based on natural language instructions. This showcases how large language models can enable automated assistants to accomplish real-world tasks. The main contributions are the novel LLMPA architecture optimized for app process automation, the methodology for applying LLMs to mobile apps, and demonstrations of multi-step task completion in a real-world environment. Notably, this work represents the first real-world deployment and extensive evaluation of a large language model-based virtual assistant in a widely used mobile application with an enormous user base numbering in the hundreds of millions.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08…$
https://doi.org/10.1145/3637528.3671646

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; • **Information systems** → *Process control systems*; • **Computing methodologies** → **Natural language processing**.

KEYWORDS

Intelligent Virtual Assistants, Agents, Large Language Models, Process Automation, Explainable

ACM Reference Format:
Yanchu Guan*, Dong Wang*, Zhixuan Chu*,†, Shiyu Wang*, Feiyue Ni, Ruihua Song, and Chenyi Zhuang. 2024. Intelligent Agents with LLM-based Process Automation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3637528.3671646

1 INTRODUCTION

In modern times, intelligent virtual assistants such as Siri, Alexa, and Google Assistant have become widespread in people's daily lives. However, despite their prevalence, these artificial intelligence systems still face constraints in their capacity to carry out intricate multi-step procedures for their human users. Nevertheless, with the swift advancement and evolution of large language models (LLMs) [4, 6, 8, 20, 28, 32, 34], there is optimism that these LLMs may help conquer the existing limitations by comprehending natural language directions more profoundly, applying logic to identify objectives, and independently orchestrating sequences of activities. By enhancing their natural language processing and reasoning abilities, large language models could enable virtual assistants to understand ambiguous instructions, break down complex goals into executable steps, and autonomously complete chained tasks to fulfill user requests [10, 14]. The continued progress in large language model research shows promise for overcoming the boundaries of today's virtual assistants.

*These authors contributed equally to this work.
†Corresponding author. The author is with the State Key Laboratory of Blockchain and Data Security & Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou, China.

This paper proposes a novel intelligent virtual assistant system based on LLM-Based Process Automation (LLMPA), which can automatically perform operations within mobile applications based on high-level user requests. Unlike prevalent virtual assistants which rely heavily on invoking fixed programmatic functions, our proposed system directly emulates detailed human interactions for carrying out tasks. This human-centric approach grants greater adaptability to perform more unconstrained, multi-stage procedures based on natural language directions. While prevailing assistants like Siri and Alexa can only execute simplistic pre-defined behaviors through rudimentary API calls, mimicking human-like actions empowers our assistant to operate at a higher level of abstraction. By simulating granular operations like clicks, scrolls, and types, our agent can flexibly conduct these operations. This enables accomplishing substantially more complex goals involving free-form instructions, creative problem-solving, and generalized tasks beyond the rigid constraints of current assistants' underlying frameworks.

The proposed LLM-Based Process Automation (LLMPA) is central to our approach, with modules for decomposing instructions, generating natural language descriptions, detecting interface elements, predicting next actions, and checking for errors. We demonstrate our system using the Alipay mobile payments app as a target environment. Users can simply describe a high-level task, such as ordering a coffee, and our system will automatically navigate the app, select items, enter information, and make payments as needed. This showcases how large language models can enable automated mobile assistants to carry out complex real-world tasks based on natural language instructions and environmental context.

The main contributions of our work are the novel LLMPA model architecture optimized for app automation, the methodology for applying LLM-based assistants to mobile apps, and demonstrations of multi-step task completion in a real-world environment. Notably, this work represents the first real-world deployment and extensive evaluation of a large language model-based virtual assistant in a widely used mobile application with an enormous user base numbering in the hundreds of millions. By successfully demonstrating complex multi-step task completion capabilities in the massively popular Alipay platform, our system marks a major milestone in translating large language model research from controlled experimental settings into large-scale practical applications with tremendous reach and impact. The expansive testing of our approach in a real production environment at this scale is unprecedented in the field of intelligent assistants, underscoring the significant advancements enabled by modern large language models toward building assistants that can reliably understand instructions, reason about goals, and accomplish procedural tasks to aid millions of end users.

2 RELATED WORK

For a while now, artificial intelligence has aimed to develop agents that possess general intelligence and can perform cognitive tasks like humans [7, 35]. Ideally, these agents should be capable of communicating through natural language and solving any computer task that a human can. For example, LangChain provides tools and abstractions to improve the customization, accuracy, and relevancy of the information the models generate.

In recent years, there have been several advancements in the field of autonomous agents as virtual assistants. For example, Apple introduced Siri, a voice assistant that helps users automate various tasks through interactive agents. Amazon launched Alexa, a virtual assistant that serves as a home automation system to control multiple smart applications or devices and perform various tasks. Google also introduced Google Assistant, an automated agent for human-computer interaction, aimed at enabling users to communicate with devices seamlessly through natural language.

It is worth noting that the combination of Large Language Models (LLMs) and Autonomous Agents has become a popular trend, largely due to the recent success of large language models. By harnessing LLMs, we can effectively handle natural language inputs and perform logical reasoning to understand user intent, enabling us to act as virtual agents for various complex tasks. This trend has greatly encouraged the exploration of LLM-augmented Autonomous Agents (LAAs) in real-world applications. For example, MindACT [12] is a web agent that can follow language commands to perform complex tasks on any website. MetaGPT [19] is an innovative framework that incorporates efficient human workflows as meta-programming methods into LLM-based multi-agent collaboration. Flowris [31] is a conversation agent focused on data analysis, which can collect and manage source data of conversation agents for data analysis and management purposes.

Nevertheless, it is imperative to acknowledge that prevailing approaches encounter numerous obstacles. The core requirement for autonomous agents is the ability to accurately understand user intent and automatically generate corresponding actions. Therefore, we present a novel approach that utilizes instruction chain technology to refine the LLM and engender manageable sequences of actions. By harnessing the formidable capabilities of LLM, we acquire profound insights into the contextual backdrop of users, thereby facilitating the prognostication of their forthcoming actions. Furthermore, cognizant of the inherent indeterminacy in generated actions, we have built a controllable calibration module to scrutinize the logical coherence of the action sequences.

3 FRAMEWORK

This section provides an overview of the proposed Intelligent Virtual Assistants. As shown in Figure 1, the user engages with the chatbot, clearly outlining the objective. The LLMPA agent collaborates with the app, aiding the user in accomplishing the operations.

The Chatbot includes a multi-turn dialogue module and an intent extraction module. In simple terms, it is designed to comprehend user requirements and generate appropriate task descriptions. Currently, a lot of research [1, 5, 23, 44] is being conducted on the topic, which is not the focus of our paper.

The LLMPA agent of our proposal is dedicated to understanding the task, deconstructing it, and then methodically executing it. This agent includes the following modules: 1) **Instruction Chains Generator.** This module decomposes the task and produces detailed step descriptions. 2) **Previous Action Description Generator.** Based on the prior action and the page content, this module produces an intelligible description [29] of the action. 3) **Object Detection.** We introduce an object detection [45] model for page section recognition. The text within a section is classified into a

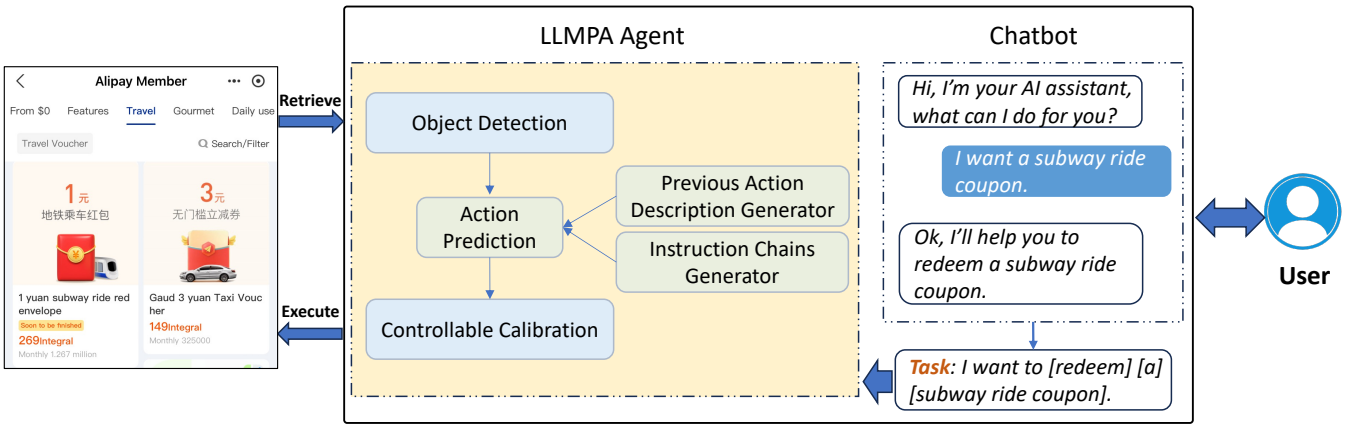

Figure 1: Architecture of Intelligent Virtual Assistants that includes LLMPA agent and Chatbot.

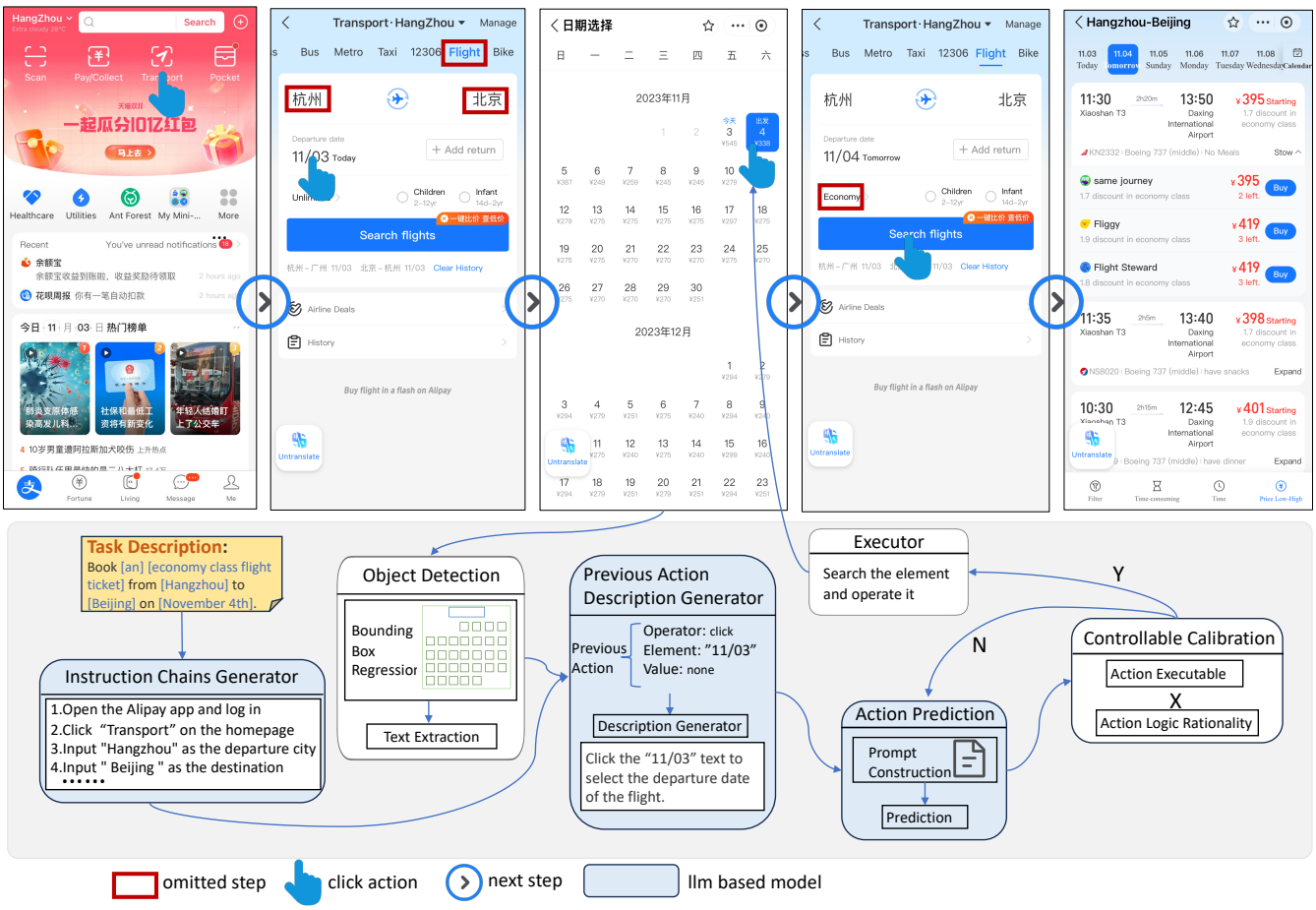

Figure 2: Pipeline of LLMPA agent.

group, which presents a clear hierarchical structure, contributing to improved comprehension of the context. 4) **Action Prediction.** Based on the output of the preceding modules, we can construct a prompt [24, 37, 38], directly predicting the subsequent action. 5) **Controllable Calibration.** Many studies [3, 13, 43] have been discussing hallucination phenomena in large language models(LLMs). To mitigate the impact of the hallucination issue, we designed a controllable calibration module. This module is utilized to scrutinize the predicted action, ensuring that the action is operable.

4 METHODOLOGY

In this section, we choose the example of "booking a flight ticket on Alipay" to illustrate how the LLMPA agent works. We will explain every module in detail, following the pipeline shown in Figure 2. Given the extensive length of the complete flight booking procedure, we display only a portion of the steps involved. Steps that have been omitted are conveniently marked with red boxes. Since the task unfolds in a loop with each step adhering to the same procedure, we elect to illustrate this process using the "select flight date" step as an example. Let's start with some notations and definitions.

4.1 Notations and Definitions

In this section, we give the following definitions used in the paper.

Definition 1. **action.** Let $\mathcal{A}$ denote the action space. For any action $a \in \mathcal{A}$, we have $a = f(e, v)$. Here, e represents the element, which can be any text on the page. v stands for value and f means the function, where $f \in \{click, scroll, type\}$. Only the type function needs a value.

Definition 2. **page content.** The page content is composed of numerous UI trees [42]. A UI tree is a structured JSON array, encompassing various attributes such as text, position, size, and color. The element of action refers to the text here. It's important to emphasize that the data from UI trees is processed through SecretFlow [26] for privacy preservation, with only generic content being retained.

Definition 3. **candidate action elements** In the process of executing the action prediction, we design a candidate set embedded within the prompt. This particular candidate set is derived from the text corpus of the page content.

4.2 Object Detection

Utilizing the raw page content to construct a prompt engenders three problems:

- The abundance of the candidate action elements amplifies the difficulty in selecting the correct answer.
- Within a page, the potential existence of identical text segments representing distinct meanings compromises the uniqueness of each element.
- Original page content consumes a large number of tokens, leading to the overly long context problem.

To alleviate the first two problems, we integrate a visual model to comprehend the page content. The information from UI trees [42] allows for a rough reconstruction of the overall page layout, thereby leveraging visual capabilities to analyze the relationships between different action elements. As shown in Figure 2, the date and the price on the calendar form a whole and together point to one meaning, i.e., selecting the flight date. Therefore, when building the candidate action elements, only one needs to be selected. So we optimize the grouping of page content through a bounding box detection model. Moreover, through effective grouping, the surrounding text [17] can endow candidate action elements with uniqueness. Specifically, we employ the YOLOX [18] model, which is an efficient and practical object detection model that maintains high accuracy. On our dataset, it achieved an excellent performance of **mAP0.92@IoU75** [18], demonstrating its powerful performance in practical applications.

To mitigate the third problem mentioned above, we designed a text extraction module that extracts text from the page content, saving a substantial number of tokens.

4.3 Previous Action Description Generator

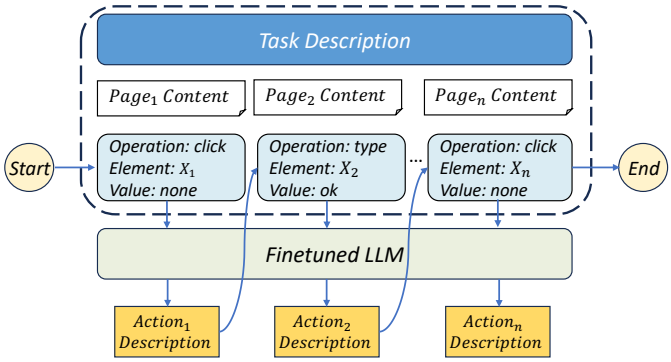

Figure 3: Structure of Previous Action Description Generator.

Through the Object Detection module, LLMPA agent is equipped to proficiently delineate the hierarchical structure of the page content. Subsequently, we will introduce another significant feature of the agent: the action-understanding capability. This feature offers two advantages:

- It defines the progress of task execution clearly.
- It bolsters the context sensitivity, implying that the action prediction model can more proficiently emphasize the influence of earlier operations on succeeding ones.

Following prior works [16], in our scenario, the automatic execution of pages can also be described as a deterministic sequential decision problem. Assuming that the present prediction corresponds to the i-th step. We define the fixed instruction as I, the prior action space as $\mathcal{A} = \{a_1, a_2, ...a_{i-1}\}$ and the page content space as $\mathcal{P} = \{p_1, p_2, ..., p_i\}$. The final function to be optimized is $g(I, A, P)$. For a specific a_j in space $\mathcal{A}$, such as clicking "ok", it may signify confirmation of items in a shopping cart, acceptance of a special deal, and so on. Essentially, it is a low-semantic command that lacks context, thereby failing to accurately convey its inherent meaning.

Figure 3 shows the structure of the Previous Action Description Generator. Based on the high-level task description and the corresponding historical behavior sequences on Alipay, the executable key paths can be excavated and represented as action sequences, which are defined as $C = \{a_1, a_2, ..., a_n\}$. $\mathcal{P}$ is the page content

space, each a_i corresponds to p_i. We adopt a recursive architecture, signifying that comprehending prior actions serves as the input for interpreting the next action. This design makes the semantic expression between continuous actions more coherent. So the function expression for the i-th action is:

$$f_i = z(a_i, p_i, p_{i+1}, f_{i-1}), \tag{1}$$

where f_i means the description of the i-th action, and z is the LLM-based model. The final output offers insight into both the immediate action behavior and the subsequent page alterations.

4.4 Instruction Chains Generator

Drawing from human cognition, we typically decompose intricate tasks [36, 39] into a series of simpler sub-tasks, allowing for a sequential execution. Inspired by this, we propose the concept of instruction chains, which generates corresponding steps according to the task description. It provides a comprehensive summary, helping the action prediction model to understand the overall process of the task more clearly and to predict the next action better.

Two kinds of instruction chains are defined:

- **Abstract Instruction Chains.** It provides a macroscopic perspective, which helps to simplify complex tasks. This allows the model to comprehend the task process and objectives from a comprehensive view. Next, we will illustrate with a concrete example:

> Task: Book an economy class flight ticket from Hangzhou to Beijing on November 4th
>
> 1. Open a reliable travel booking platform.
> 2. Enter "Hangzhou" as departure, "Beijing" as destination, and select November 4th.
> 3. Choose "Economy Class" and search for flights.
> 4. Select your preferred flight and proceed to book.
> 5. Fill in the necessary traveler details and go to payment.
> 6. Complete the payment and receive a confirmation email.

- **Elaborate Instruction Chains.** It offers detailed guidance, minimizing confusion and ambiguity. As a result, the model is able to predict subsequent actions more easily. Similarly, we will demonstrate this with an example:

> Task: Book an economy class flight ticket from Hangzhou to Beijing on November 4th
>
> 1. Open the Alipay app and log in.
> 2. Click "Transport" on the homepage.
> 3. Input "Hangzhou" as the departure city.
> 4. Input "Beijing" as the destination.
> 5. Set the date to November 4th.
> 6. Select the "Economy" class flight.
> 7. Search flight and choose an appropriate flight.

The most significant distinction between the two examples above lies in the latter's tighter integration with the action space. Naturally, the latter also demands a greater quantity of high-quality samples in the relevant field. To ensure precise execution of tasks, we trained the elaborate instruction chains generation model in our scene. Just like the previous action description generation model, this model is also based on LLM. In contrast, we utilize the abstract instruction chains in the subsequently introduced Agentbench-WB [25] dataset.

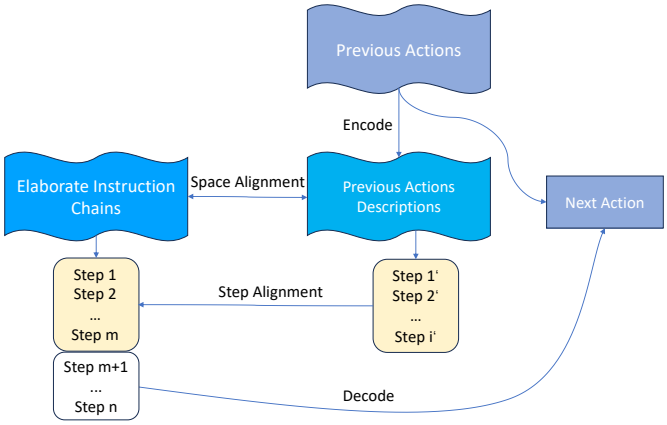

Figure 4: Advantages of Combining Action Descriptions and Elaborate Instruction Chains.

Upon analysis, the combination of Previous Action Description(PAD) and Elaborate Instruction Chains(EIC) can significantly reduce the difficulty of the next action prediction. As shown in Figure 4, different colors represent different spaces, and the closer the colors are, the higher the spatial similarity. PAD can be characterized as the encoding process from action to action description. Since the generation of EIC highly depends on the action, it can be approximated that EIC and PAD are spatially similar. Assuming a task is given and has been executed for the i-th step. We define the space of EIC as $\mathcal{S} = \{Step_1, Step_2, ..., Step_n\}$, and the space of PAD as $\mathcal{S}' = \{Step'_1, Step'_2, ..., Step'_i\}$. By calculating semantic similarity, S' can be aligned to a continuous subspace of $\mathcal{S}$, assuming this space is $\mathcal{S}_\alpha = \{Step_1, Step_2, ..., Step_m\}$, then we have $\mathcal{S}' \simeq \mathcal{S}_\alpha$. Therefore, the prediction space for the next step will be narrowed down to $\mathcal{S}_\beta = \{Step_{m+1}, ..., Step_n\}$. This yields explicit assistance for subsequent action prediction.

4.5 Action Prediction

As illustrated in Figure 5, our prompt is an amalgamation of several components: task description, instruction chains, history actions, descriptions of history actions, extracted text, and candidate action elements. The candidate action elements are derived from the top elements of historical data, the task itself, and the extracted text from bounding boxes. We then employ the LLM to train our next-action prediction model.

4.6 Controllable Calibration

Due to the hallucination problem of large language models(LLMs), even given a candidate, it may still generate other answers. We designed the controllable calibration module, which is composed of two parts:

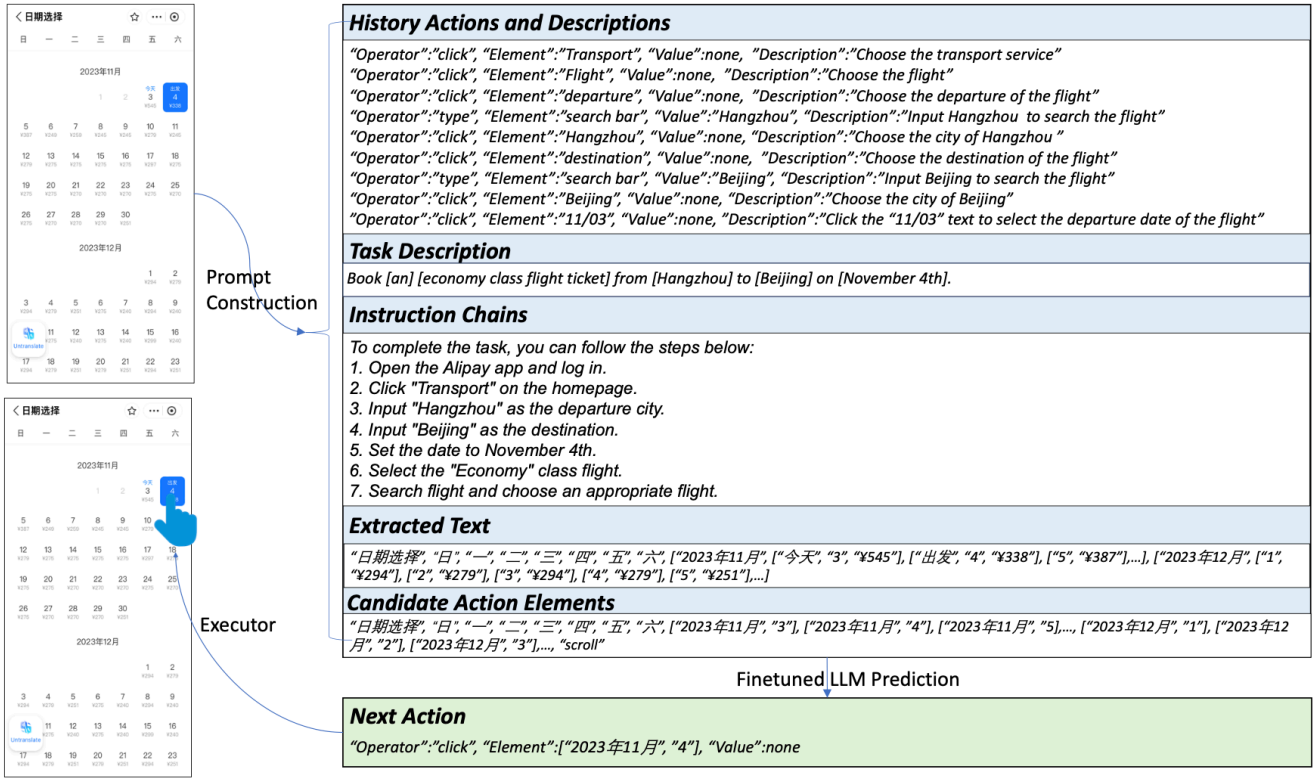

Figure 5: An example flow with history actions, descriptions of history actions, task description, instruction chains, extracted text, and candidate action elements in the real Alipay scene of flight booking.

- **Is it executable?** A CTR model [21] can be trained to verify whether the predicted element can be executed. The element can be treated as an item. If a click or type occurs, it is labeled as a positive sample. This transforms the problem into a classic binary classification problem [22]. By setting a threshold, we can determine whether the element is executable.
- **Does it make logical sense?** Based on the executable key paths mentioned in the Previous Action Description Generator, we add a constraint. If there is no loop in the original path, it is logically unreasonable for a loop to appear in the result.

If the predicted element passes both checks successfully, it will be handed over to the executor for simulation-based operations. However, in case of failure, it will be rerouted back to the Action Prediction module for further output revision.

5 EXPERIMENT

To evaluate the effectiveness of the LLMPA agent, we conduct experiments on both real-world online environments within Alipay and public benchmarks.

5.1 Online Deployment

For online evaluation, we validate the effectiveness of our framework with Alipay.

Environment. Alipay, a prominent global payment tool, encompasses numerous services that cater to various digital life and finance scenarios. Our proposed LLMPA agent has significantly enhanced the user experience. We have covered more than 20 real-world scenarios in online deployment and manually annotated 2000 different tasks. Extensive online experimentation has effectively demonstrated the superiority of our method.

Setting. It should be noted that AntLLM-10b is a pretrained large language model in our work, which is trained from scratch with the GLM [15, 40] structure. We report the performance of AntLLM-10b that serves as a baseline and LLMPA agent that integrates components including *Object Detection*, *Instruction Chains & Previous Action Description*, and *Controllable Calibration*. Regarding *Instruction Chains*, Elaborate Instruction Chains are leveraged considering that in the online environment, we have access to rich and detailed resources of action sequence. We also conducted an ablation study to evaluate the impact of different components on the performance of the LLMPA agent. The models above are fine-tuned during online experiments.

Metric. We report Success Rate as the evaluation metric including *Step Success Rate (Step SR)* and *Task Success Rate (Task SR)*. A step is considered successful when the selected element and the predicted action correspond to the ground truth. A task is considered successful when all of its constituent steps have succeeded.

Deployment details. Our method was trained using 32 NVIDIA A-100 GPUs. For online inference, we employed 100 NVIDIA A-40 GPUs in the production environment. We conducted weekly finetuning using the latest data to ensure the freshness of model performance.

Table 1: Online experimental results with Alipay environment. *Step SR* and *Task SR* stands for *Step Success Rate* and *Task Success Rate,* while *IC & PAD* stands for *Instruction Chains & Previous Action Description. Baseline* means finetuned AntLLM-10b.

Method	Step SR	Task SR
Baseline	52.42	6.05
LLMPA w/o Object Detection	86.02	48.46
LLMPA w/o IC & PAD	65.43	14.27
LLMPA w/o Controllable Calibration	84.53	45.85
LLMPA	**93.71**	**70.42**

Online Performance and Analysis. As shown in Table 1, the AntLLM-10b without any additional design obtains the poorest performance as expected. LLMPA outperforms the baseline by a significant margin and achieves a high success rate in terms of step success rate (93.71%), demonstrating the effectiveness of the framework. Furthermore, the absence of each component leads to a noticeable decrease in performance, with *Instruction Chains & Previous Action Description* having the most significant impact. This indicates the importance of task decomposition and high-level summarization capabilities.

5.2 Benchmark Test

Table 2: Experimental results on public benchmark AgentBench-WB. *Step SR*, *Ele. Acc*, and *Op. F1* stands for *Step Success Rate, Element Accuracy* and *Operation F1. IC* stands for *Instruction Chains.*

Models	IC	Step SR	Ele. Acc	Op. F1
`llama2-7b-chat [33]`	✗	7.25	9.38	44.71
	✓	8.10	9.55	45.76
`llama2-13b-chat [33]`	✗	9.29	10.40	47.36
	✓	10.27	11.96	47.52
`baichuan2-7b-chat [2]`	✗	10.49	10.91	48.22
	✓	11.51	13.81	46.79
`gpt-3.5-turbo [27]`	✗	16.79	21.23	47.45
	✓	17.82	22.51	45.09
`gpt-4 [28]`	✗	24.65	33.47	52.94
	✓	26.36	35.23	52.65

We conducted experiments on public datasets to evaluate the effectiveness of the LLMPA.

Environment. We select the Web Browsing environment from AgentBench (namely AgentBench-WB) [25] as the representative benchmark. AgentBench-WB, derived from Mind2Web dataset [12], encompasses 912 tasks from 73 websites that span domains such as Housing, Job, Social Media, Education, etc.

Meanwhile, we have also conducted evaluations on the AITW dataset [30], which encompasses a wide array of multi-step tasks such as web and app operations, and app installations. It includes 715k episodes and 15k unique prompts.

Setting. Evaluation on AgentBench-WB[25] involves two stages: ranking HTML elements with a fine-tuned small language model and predicting action in the form of multi-choice QA with an agent. The ranking models employed in our experiments align with AgentBench, and the primary focus of offline experiments is to evaluate the capability of action prediction. It is worth noting that there still remains a gap between the simulation environment and real-world scenarios in this area. For instance, in comparison to the online environment, AgentBench-WB offers a relatively limited number of web pages and lacks dynamic interaction capability. Consequently, it is not feasible to implement all of the methods we proposed in the simulation environment. Therefore, we conduct the evaluation solely focusing on the effectiveness of *Instruction Chains*, which is proved to be a core component of LLMPA. We employ *Instruction Chains* in the form of Abstract Instruction Chains, for the limited resource of action sequences in a simulation environment. Evaluations are conducted in the manner of in-context learning.

We also conducted similar experiments on the AITW dataset [30]. Based on the Auto-UI model [41], we incorporated instruction chains and previous action descriptions and then retrained the model.

Metric. *Step Success Rate* is employed as an evaluation metric. Following Mind2Web[12], we also report *Element Accuracy* that calculates the accuracy of the selected element, and *Operation F1* that calculates the token-level F1 score for the predicted operation.

Following Auto-UI [41], we use *action success rate* as the evaluation metric on the AITW dataset [30].

Implementation details. All experiments of benchmark testing run on the Linux server(Ubuntu 16.04) with the Intel(R) Xeon(R) Silver 4214 2.20GHz CPU, 512GB memory, and 8 NVIDIA A-100 GPUs.

Results. From the results presented in Table 2, we observe that the leverage of *Instruction Chains* contributes to enhancing performance on both open-sourced and API-based models, suggesting the effectiveness and generalization capability of the method across different models. Meanwhile, in comparison to online experiments, the improvements brought by *Instruction Chains* are less pronounced, indicating that more specific instruction chains lead to higher performance. In addition, Table 3 shows the main results of the AITW dataset. Our model demonstrates the best performance compared to all other baselines. This also proves that our proposed Instruction Chains and Previous Actions Descriptions can better assist the model in reasoning, and completing tasks more efficiently.

Table 3: Experimental results on public benchmark AITW. *action success rate* is the evaluation metric. The results of BC and PaLM-CoT are from [30]. The results of ChatGPT-CoT, Llama2, and Auto-UI are from [41]. Our model is based on $Auto-UI_{unified}$, enhanced with Instruction Chains and Previous Actions Descriptions. The bold numbers represent the best results.

Models	Overall	General	Install	GoogleApps	Single	WebShopping
BC-single	68.7	-	-	-	-	-
BC-history	73.1	63.7	77.5	75.7	80.3	68.5
PaLM 2-CoT	39.6	-	-	-	-	-
ChatGPT-CoT	7.72	5.93	4.38	10.47	9.39	8.42
Fine-tuned Llama 2	28.40	28.56	35.18	30.99	27.35	19.92
$Auto-UI_{separate}$	74.07	65.94	77.62	**76.45**	81.39	69.72
$Auto-UI_{unified}$	74.27	68.24	76.89	71.37	84.58	70.26
Ours	**76.52**	**69.78**	**79.61**	73.82	**86.86**	**72.53**

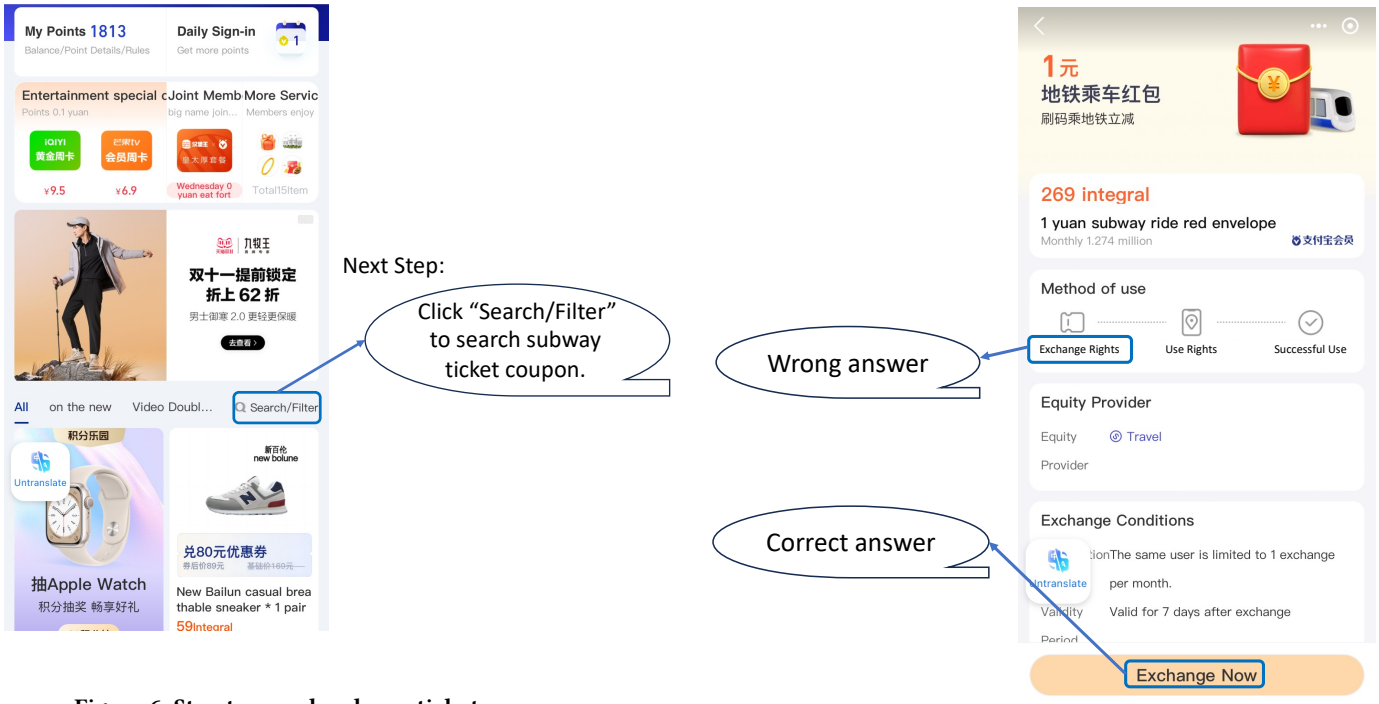

Figure 6: Step to search subway ticket coupon.

Figure 7: Step to exchange subway ticket coupon.

5.3 Case Study

To validate the effectiveness of our proposed methodology, we analyze a practical application, specifically, the process of redeeming subway discount vouchers in the Alipay membership scenario. Figure 6 illustrates one step of the full process where normally expect to click the search bar and type subway ticket coupon. It is significantly difficult for the model to infer the underlying significance of this search bar by relying solely on the contextual information, leading to incorrect predictions. From a certain perspective, instruction chains could be regarded as an invaluable enhancement of external knowledge, which offers pivotal guidance for the action prediction model to comprehend the task's inherent operational logic.

As depicted in Figure 7, the accurate action entails clicking on "Exchange Now". However, the action prediction model erroneously returns "Exchange Rights". Even though these two elements bear a high degree of semantic similarity, they differ significantly in terms of operability. In such instances, the controllable calibration module is adept at discerning that "Exchange Rights" is non-clickable and non-typable. Upon receiving this feedback, the model refines its output accordingly. Through this mechanism, the model can effectively sidestep a host of glaring errors.

6 DISCUSSION

The proposed virtual assistant system based on large language models shows promise in its ability to parse complex instructions, reason about goals, and execute chained tasks autonomously. However, there are both advantages and disadvantages to this approach that warrant further discussion.

A key advantage is the enhanced natural language processing and reasoning capabilities enabled by the large language model architecture of the LLMPA module. By leveraging large amounts of training data, the LLM is able to comprehend ambiguous or incomplete instructions and better infer user intent. This allows the assistant to successfully interpret and act on a wider range of natural language requests. Additionally, the LLM's ability to break down goals and predict the next actions facilitates multi-step procedural task completion.

However, there are also limitations to relying solely on the LLM. The system remains constrained by the training data, meaning unfamiliar phrasings or requests may still pose a challenge. There are also risks of biased or erroneous behavior if the training data contains imperfections. From an implementation standpoint, large language models can be resource-intensive, making deployment on mobile devices challenging. More work is needed to optimize the models for on-device usage.

The capabilities enabled by large language model virtual assistants also raise important ethical considerations regarding potential misuse. There is a risk of these systems being leveraged to spread misinformation or manipulate users [9, 11]. Continued research is critical to develop safeguards against harmful applications. In addition, with any software involving personal data, maintaining user privacy is paramount. Strict access controls and encryption should be implemented to secure information flows throughout the system architecture. Only essential data should be retained, with procedures to delete sensitive user inputs after task completion. Consent procedures must transparently communicate how user data will be handled. Compliance with relevant privacy legislation and best practices around data minimization and anonymization should be prioritized during development and deployment. Independent audits can validate privacy preservation mechanisms.

7 CONCLUSION

This work introduced a novel approach to intelligent virtual assistants using large language models designed specifically for mobile app automation. We proposed an end-to-end architecture consisting of the LLMPA model, environmental context, and an executor that enabled automated multi-step task completion in a real-world payment app based on natural language instructions. Testing at a large scale in the widely used Alipay platform demonstrated that modern large language models can power assistants capable of understanding goals, planning, and accomplishing intricate real-world procedures to assist users. This represents a major advance for intelligent assistants and their adoption in ubiquitous mobile applications. Further development of contextual processing, reasoning capabilities, and optimized on-device deployment can build on these foundations to realize the full potential of virtual agents able to comprehend language, plan, and take action to aid humans in their daily lives.

REFERENCES

[1] Eleni Adamopoulou and Lefteris Moussiades. 2020. An overview of chatbot technology. In *IFIP international conference on artificial intelligence applications and innovations*. Springer, 373–383.

[2] Baichuan. 2023. Baichuan 2: Open Large-scale Language Models. *arXiv preprint arXiv:2309.10305* (2023). https://arxiv.org/abs/2309.10305

[3] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023* (2023).

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Jack Cahn. 2017. CHATBOT: Architecture, design, & development. *University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science* (2017).

[6] Zhixuan Chu, Huaiyu Guo, Xinyuan Zhou, Yijia Wang, Fei Yu, Hong Chen, Wanqing Xu, Xin Lu, Qing Cui, Longfei Li, Jun Zhou, and Sheng Li. 2023. Data-Centric Financial Large Language Models. arXiv:2310.17784 [cs.CL]

[7] Zhixuan Chu, Hongyan Hao, Xin Ouyang, Simeng Wang, Yan Wang, Yue Shen, Jinjie Gu, Qing Cui, Longfei Li, Siqiao Xue, et al. 2023. Leveraging large language models for pre-trained recommender systems. *arXiv preprint arXiv:2308.10837* (2023).

[8] Zhixuan Chu, Yan Wang, Qing Cui, Longfei Li, Wenqing Chen, Sheng Li, Zhan Qin, and Kui Ren. 2024. Llm-guided multi-view hypergraph learning for human-centric explainable recommendation. *arXiv preprint arXiv:2401.08217* (2024).

[9] Zhixuan Chu, Yan Wang, Longfei Li, Zhibo Wang, Zhan Qin, and Kui Ren. 2024. A Causal Explainable Guardrails for Large Language Models. *arXiv preprint arXiv:2405.04160* (2024).

[10] Zhixuan Chu, Yan Wang, Feng Zhu, Lu Yu, Longfei Li, and Jinjie Gu. 2024. Professional Agents–Evolving Large Language Models into Autonomous Experts with Human-Level Competencies. *arXiv preprint arXiv:2402.03628* (2024).

[11] Zhixuan Chu, Lei Zhang, Yichen Sun, Siqiao Xue, Zhibo Wang, Zhan Qin, and Kui Ren. 2024. Sora Detector: A Unified Hallucination Detection for Large Text-to-Video Models. *arXiv preprint arXiv:2405.04180* (2024).

[12] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2Web: Towards a Generalist Agent for the Web. *arXiv preprint arXiv:2306.06070* (2023).

[13] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495* (2023).

[14] Xin Luna Dong, Seungwhan Moon, Yifan Ethan Xu, Kshitiz Malik, and Zhou Yu. 2023. Towards Next-Generation Intelligent Assistants Leveraging LLM Techniques. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5792–5793.

[15] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[16] Hiroki Furuta, Ofir Nachum, Kuang-Huei Lee, Yutaka Matsuo, Shixiang Shane Gu, and Izzeddin Gur. 2023. Multimodal Web Navigation with Instruction-Finetuned Foundation Models. *arXiv preprint arXiv:2305.11854* (2023).

[17] Bin Gao, Tie-Yan Liu, Tao Qin, Xin Zheng, Qian-Sheng Cheng, and Wei-Ying Ma. 2005. Web image clustering by consistent utilization of visual features and surrounding texts. In *Proceedings of the 13th annual ACM international conference on Multimedia*. 112–121.

[18] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. 2021. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430* (2021).

[19] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352* (2023).

[20] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728* (2023).

[21] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM conference on recommender systems*. 43–50.

[22] Roshan Kumari and Saurabh Kr Srivastava. 2017. Machine learning: A review on binary classification. *International Journal of Computer Applications* 160, 7 (2017).

[23] Chi-Hsun Li, Su-Fang Yeh, Tang-Jie Chang, Meng-Hsuan Tsai, Ken Chen, and Yung-Ju Chang. 2020. A conversation analysis of non-progress and coping strategies with a banking task-oriented chatbot. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[24] 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. *Comput. Surveys* 55, 9 (2023), 1–35.

[25] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688* (2023).

[26] Junming Ma, Yancheng Zheng, Jun Feng, Derun Zhao, Haoqi Wu, Wenjing Fang, Jin Tan, Chaofan Yu, Benyu Zhang, and Lei Wang. 2023. {SecretFlow-SPU}: A Performant and {User-Friendly} Framework for {Privacy-Preserving} Machine Learning. In *2023 USENIX Annual Technical Conference (USENIX ATC 23)*. 17–33.

[27] OpenAI. 2022. Introducing chatgpt. (2022).

[28] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[29] Peng Qin, Weiming Tan, Jingzhi Guo, and Bingqing Shen. 2021. Intelligible description language contract (IDLC)–A novel smart contract model. *Information Systems Frontiers* (2021), 1–18.

[30] Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Android in the wild: A large-scale dataset for android device control. *arXiv preprint arXiv:2307.10088* (2023).

[31] Jiajia Sun, Juan Wang, Yueguo Chen, and Xiongpai Qin. 2023. Flowris: Managing Data Analysis Workflows for Conversational Agent. In *International Conference on Database Systems for Advanced Applications*. Springer, 724–728.

[32] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[33] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[34] Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Zhang, Qing Cui, et al. 2024. LLMRG: Improving Recommendations through Large Language Model Reasoning Graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 19189–19196.

[35] Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Y Zhang, Qing Cui, et al. 2023. Enhancing recommender systems with large language model reasoning graphs. *arXiv preprint arXiv:2308.10835* (2023).

[36] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[37] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382* (2023).

[38] Siqiao Xue, Yan Wang, Zhixuan Chu, Xiaoming Shi, Caigao Jiang, Hongyan Hao, Gangwei Jiang, Xiaoyun Feng, James Y Zhang, and Jun Zhou. 2023. Prompt-augmented temporal point process for streaming event sequence. *arXiv preprint arXiv:2310.04993* (2023).

[39] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601* (2023).

[40] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414* (2022).

[41] Zhuosheng Zhan and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents. *arXiv preprint arXiv:2309.11436* (2023).

[42] Xiaoyi Zhang, Lilian de Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, et al. 2021. Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–15.

[43] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *arXiv preprint arXiv:2309.01219* (2023).

[44] Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences* 63, 10 (2020), 2011–2027.

[45] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2023. Object detection in 20 years: A survey. *Proc. IEEE* (2023).