SATBAYEV UNIVERSITY

Computing & Engineering

Volume 2 (2024), Issue 3, 1-6

https://doi.org/10.51301/ce.2024.i3.01

Self-correcting generative software for prompt processing

K. Urazov*, A. Razaque, Zh. Kalpeyeva

Satbayev University, Almaty, Kazakhstan

**Corresponding author: kaisarkz347@gmail.com*

Abstract. Large language models have difficulty understanding the context, have discrepancies with the task at hand, and quite often make mistakes. A person is forced to interfere quite often in AI work, which is ineffective for large and time- consuming tasks. This study suggests the following: AI teams and coordination of their work, which interact with each other to achieve a set goal. In this study, this will be achieved by passing messages between AI through plain text. The implementation includes using the API of a popular messenger to wrap the AI into a certain bot, which will be assigned certain roles. Each such bot will be assigned its roles and requests, and they, guided by their role, request, and task, will jointly solve the task assigned to the group. The expected results are a significant reduction in human intervention, increased automation, system performance and reliability, and a wider range of applications.

Keywords: *large language model, multi-agent LLM collaboration, AI-powered systems.*

1. Introduction

Recently, large language models have shown outstanding results. The transformer-based architecture makes it possible to better process natural language and generate it [1]. But despite this, it is still not enough to call them impeccable, they may misunderstand the context, give information that does not exist, mislead and miss details, and, most importantly, they cannot correct themselves. These shortcomings require human intervention to identify and correct them, which is ineffective [2].

Dependence on humans generates the human factor, which leads to unforeseen errors and slows down the overall automation process. But what if such a system is used in finance or healthcare? Even minimal errors are fundamentally unacceptable there.

To solve such problems, a different approach is proposed: teams of large language models working together [2,3]. Unlike traditional methods, our approach involves the creation of AI groups that collaborate on a task in real-time. In this way, you can reduce the likelihood of errors by correcting each other. This method is aimed at improving the reliability, ac- curacy, and efficiency of artificial intelligence-based problem- solving systems.

The proposed system uses the popular messenger application as a communication platform where a human operator can enter tasks and interact with the AI team. The server infrastructure will handle API messaging and manage the execution of multiple LM instances, each assigned specific roles and system requests. This setup simulates natural human communication, facilitating seamless interaction and coordi- nation between artificial intelligence agents.

This article describes the limitations of modern AI models, reviews literature reviews on existing approaches, and describes the architecture and implementation of the proposed self-correcting AI system. Demonstrating the effectiveness of multi-agent AI teams.

1.1. Problem statement

Despite the advancements of Language Models (LMs) in understanding language and completing tasks, they still have limitations: they can misunderstand context, give wrong information, leave out important details, and fix their mistakes automatically. It takes a lot of work, time, and care to rely only on people to fix these mistakes. This can lead to unexpected behavior. As AI systems take on more complex tasks, relying on humans to manually fix errors is unreliable. This dependence on humans increases the probability of errors. In critical areas, such mistakes can lead to severe consequences. There are limits to what a single AI can do, so building teams of AI to fix their own mistakes could be a useful idea. These teams would be made up of LMs that work together in real-time to find and fix errors on their own. By letting AI systems talk to each other and work together, self-correcting AI teams could improve AI-powered systems and make them much better at doing tasks, for example, answering questions with given patterns without lowering the details. This research focuses on studying how self-correcting AI teams can be used effectively in different situations where tasks need to be solved.

1.2. Related works

This section discusses the features of the existing works. A notable contribution to this field is the work of Saunders et al., which presents a self-critical framework for large language models [3]. Their method fine-tunes models to obtain natural language criticisms using behavioral cloning, which greatly helps evaluators identify weaknesses that might otherwise be overlooked. These criticisms are effective at detecting errors in both model-created and human-written summaries, including intentionally misleading content.

Saunders et al. solve key problems related to understanding the context and cooperation between humans and artificial intelligence, allowing models to repeatedly analyze and

improve their results [3]. Larger models in their structure have an increased ability to self-criticize and integrate feedback into their results, which allows for more accurate and contextually relevant results. Their work demonstrates that these opportunities for self-criticism not only improve the quality of products but also reduce dependence on direct human control, which makes it possible to expand the scope of control for tasks that are difficult for a person to assess directly. This understanding highlights the potential for further improvement of self-assessment mechanisms in artificial intelligence systems.

1.3. Literature review

LMs became more developed with transformer architectures such as GPT [1] and BERT. These models use self-attention techniques to better understand the context, allowing for more fluent text generation. LMs expand AI's ability to process and communicate with humans through language-based applications [4].

Language models (LMs) can mislead the guideline or not follow the pattern, which leads to mistakes and incorrect responses. This is because LMs could not recognize some details, and they could not handle modifiers correctly [5]. Additionally, LMs lack knowledge about a specific topic to generate accurate responses. Consequently, errors in the early stages of processing can lead to even more serious errors later, making it difficult for LMs to complete tasks requiring accuracy. Current methods for correcting errors in LMs often rely on humans to identify and fix errors, which is unreliable [2,6].

OpenAI made a showcase of their new feature – running Python code within their most advanced model [1]. This makes the interaction between LLM and practical implementation possible with rich Python libraries. Using libraries like NumPy, pandas, scikit-learn, TensorFlow, and PyTorch, GPT-4 can execute tasks such as data analysis, machine learning, web scraping, asking an API for information, and automation. This integration increases AI capabilities, offering users a tool for executing computational tasks. It can be used as an assistant in various laboratories [7].

Recent studies show that single LLM performance is lower than multiple LLM agents working together in a team [2,8,9,10,3].

1.4. Identified gaps and limitations of current approaches

Other approaches use one large language model, which, as mentioned above, is unreliable. Again, a person is forced to correct mistakes on their own and then direct a large language model more precisely, not to mention that during the generation of the answer, the AI may omit some details.

Another approach involves the use of several smaller language models wrapped in one, but then again, using even a language model as a whole can lead to unexpected consequences. The main difference from the previous method is that at the processing stage, the most qualified model is selected inside this convolution, which will be able to process the incoming information best.

1.5. Our plan and expected result

We plan to write software where LM can interact and coordinate with each other by seeing messages from each other. For proxy between a human operator and group chat, we chose a popular messenger app, where the operator can call and state the task for the AI team in messenger. Then, by API, the input is transferred to AI. A group of AI processes the input, generating the final output by coordinating with each other and providing that output back to the messenger via API. There will be a backend app that processes the API messaging and running instances of LM. Each AI output is represented as a member of the group chat in the messenger, every message of LM will be reflected in the messenger's UI. There will be a proxy in the team which accepts the input and sends it to the group. Every LM has a role and system prompt to which it will try to stick. Every LM is aware of the existence of other AIs and their roles. Then, one by one, every AI talk to the group, and the proxy AI decides to whom that output was addressed and sends it appropriately. AI can also have other group chats to talk, and two separate group chats cannot see messages from each other. Also, every AI can have hard-coded functions to call if there is a need, for example, loading information from an external API so it will not invent its link.

Shortly, this architecture is much like the natural internet communication of humans in messenger apps, with multiple chats and going for links, etc. We expect decent results.

1.6. Proposed Solution

The proposed solution is a collaboration platform in which several large language models (LLM) interact, analyze, and refine each other's results to improve the accuracy and efficiency of problem-solving. Key components of the solution include task decomposition, automated self-correction mechanisms, and a coordinated workflow.

Self-Correcting LLM Collaboration - a system S ={L1, L2, ..., Ln} where Li represents an LLM instance, n is the total number of models, and the output O is refined iteratively through peer critique:

$$O = F(I, L_1, L_2, ..., L_n),$$

where I is provided by the human and F is the mapping function that ensures correction through collaboration.

Corollary 1 The output Oi of a refinement loop converges to an optimal solution O∗ after k iterations.

Proof Assume that the initial input has a quality $1 - E0$ given that $0 < E0 < 1$. After each iteration i, the error coefficient is reduced by a correction factor α such that:

$$E_{i+1} = \alpha E_i, 0 < \alpha < 1$$

By iterating this process k times, the error approaches zero:

$$\lim_{k \to \infty} E_k = 0$$

There is a loop where one or several responsible LLMs write code until the critic agent is not satisfied with the output. Algorithm 1 illustrates the use of a collaborative mechanism for the group.

Hypothesis 1 The accuracy of task-solving increases when multiple LLMs collaborate through critique and self-correction mechanisms compared to a single LLM.

Proof. Assumptions:

• Each LLM Li has an independent probability p of solving the task correctly.

• Errors produced by one LLM can be detected and corrected by others through peer critique.

For a single LLM L_1, the probability of an error is:

$$P(E_1) = 1 - p$$

In a collaboration setting with n LLMs working independently and providing peer review, the probability that all LLMs fail to detect an error is:

$$P(E_n) = (1 - p)^n$$

Therefore, the probability that at least one LLM detects and corrects the error is the complement:

$$P_{\text{corrected}} = 1 - (1 - p)^n$$

As n increases, $P_{corrected}$ approaches 1, proving that collaborative LLM frameworks improve response. This shows that the accuracy gain is noticeable compared to a single LLM.

Proposed Architecture

The proposed system combines a collaboration platform into a messaging platform that serves as an intermediary between the human and the LLM group. Key components include:

• Input layer - messaging platform

• Processing layer -

– The input is passed to multiple LLM instances $\{L_1, L_2, \ldots, L_n\}$.

– LLMs critique and refine each other's outputs iteratively

• Output layer - The final corrected output O∗ is returned to the operator through the messaging platform.

Algorithm-1 titled «An algorithm of prompt processing with four agents», begins by taking user input (I) and transferring it to the proxy agent (A_p). Before this, a system prompt (role) has been assigned to each of the agents (A_{co}, A_{cr}, A_{pl}). Then input is passed to the group chat, and then, according to their roles, agents choose a response queue. Here planner (A_{pl}) suggests a plan A_{pld} for coder (A_{co}), it can also suggest running user-defined functions for gathering data from the internet or for other tasks. Then coder (A_{co}) generates code and proxy A_p runs it, checking if code ran successfully, but if code fails to run with success, coder (A_{co}) generates code again according to the error output. Then critic agent (A_{cr}) evaluates output (A_{cod}) of the code and decides (A_{crd}). If the decision is positive, output (R) is generated from the critic (A_{cr}).

Algorithm 1 An algorithm of prompt processing with four agents

Input: {I}

Output: {R}

Initialization: {I: Input; A_p: Proxy agent; A_{co}: Coder agent; A_{cr} : Critic agent; A_{pl}: Planner agent; A_{pld}: Planner agent's data (plan); A_{cod}: Coder agent's data (code); A_{crd}: Critic agent's data (review)}

Pass I to group chat

Set $A_{pld} \in I \rightarrow A_{pld}$

A_{crd} is negative **do**

Do process $A_{cod} \in I, A_{pld} \rightarrow A_{cod}$

if A_{cod} runs with error **then**

Rewrite A_{cod}

end if

Do process $A_{crd} \in A_{cod} \rightarrow A_{crd}$

end while

Set $R \in A_{cod} \rightarrow R$

if R is satisfactory **then**

R is completed

else R needs improvement

end if

2. Materials and methods

A survey will be conducted to evaluate the effectiveness of the proposed self-correcting artificial intelligence system and compare its performance with that of existing language models (LMS). The survey will focus on evaluating answers to a number of logical questions that require accurate understanding and context for correct answers. Human participants will evaluate the generated responses based on certain criteria. There will be a set of logic questions that will be passed to the AI to evaluate the capabilities of every system. These questions are specially selected to test the AI's ability to follow complicated instructions and provide answers. As soon as the set of questions is finalized, the answers will be obtained using separate modern LMS, such as GPT-3, GPT-4, and other well-known models, as well as the proposed multi-agent artificial intelligence system. A multi-agent artificial intelligence system based on collaboration and error correction principles will handle the same set of issues as individual LMS.

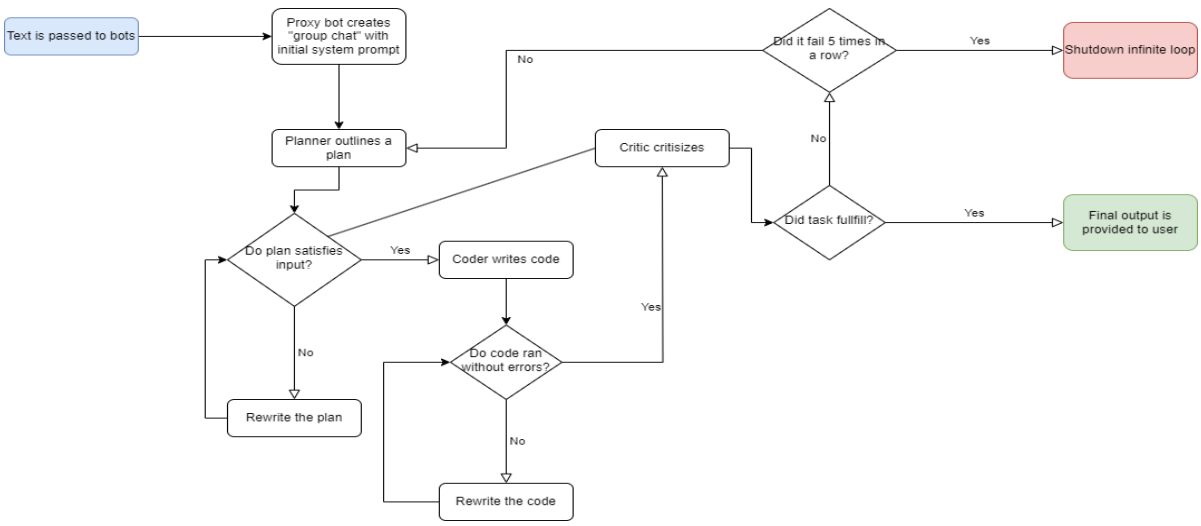

Figure 1. Event Flow Diagram

Then there will be a survey to evaluate the responses. A group of participants will be recruited to provide a sample of opinions. The survey will be conducted on an online platform where participants will be presented with contextual questions and answers from different systems. The evaluation criteria will focus on the overall quality of responses.

The results will be presented in the form of tables and graphical visualizations that represent the performance indicators of various multi-agent artificial intelligence systems. This approach will provide an objective analysis of the effectiveness of the proposed system. The purpose of the study is to demonstrate the potential advantages of a self-correcting artificial intelligence system.

A. Evaluation of Different Language Models

To evaluate the effectiveness of various language models (LMS) in solving various tasks, we conducted a series of tests on several popular models and a multi-agent artificial intelligence system proposed by us. The tasks were designed to evaluate LMS capabilities in coding, creative writing, logical thinking, and structured data creation. The results are presented in the table below, where the success of each task is indicated by a «+» sign in case of success, «-» in case of failure, and «n/a» in case of lack of access. This study will present ready-made test results from external sources for a comprehensive performance analysis. Specifically, we will utilize test results from the YouTube channel 'Matthew Berman' [11], which is featured on the LM Leaderboard [12]. The channel provides a range of prompts that are used to benchmark the performance of different LMs.

3. Results and discussion

Test Prompts

· Write a Python script to output numbers from 1 to 100.

· Write the game «snake» in Python.

· Write a poem about AI with exactly 50 words.

· Write an email to my boss letting them know I am leaving the company.

· If we lay 5 shirts out in the sun and it takes 4 hours to dry, how long would 20 shirts take to dry? Explain your reasoning step by step.

· Jane is faster than Joe. Joe is faster than Sam. Is Sam faster than Jane? Explain your reasoning step by step.

· There are three killers in a room. Someone enters the room and kills one of them. Nobody leaves the room. How many killers are left in the room? Explain your reasoning step by step.

· Create JSON for the following: There are 3 people, two males. One is named Mark. Another is named Joe. And a third person, who is a woman, is named Sam. The woman is age 30 and the two men are both 19.

B. Analysis

1. LLaMA 2 13b fp16: Demonstrated consistent performance in basic programming and creative tasks but struggled with more complex coding and logical reasoning tasks.

2. Mixtral: Demonstrated robust capabilities across all tested tasks, excelling particularly in logical reasoning and structured data creation.

3. gpt4o: Performed well in logical reasoning tasks but was not tested on all creative and communication tasks.

4. LLaMA 3: Similar to LLaMA 2, it performed well in ba- sic and some logical tasks but failed in formal communication tasks.

5. Gemini 1.5 Pro: Excelled in logical reasoning and structured data tasks but was not tested in several creative and communication tasks because of its clunkiness, and its realization, but 1 million context windows is an outstanding feature that capable of analyzing videos and enormous texts.

6. Multi-Agent AI: Multi-agent AI solution performed de- cently across all tasks. By leveraging the strengths of multiple LMs, the system was able to generate accurate, detailed, and contextually appropriate responses consistently. This demon- strates the effectiveness of the collaborative approach in over- coming individual model limitations.

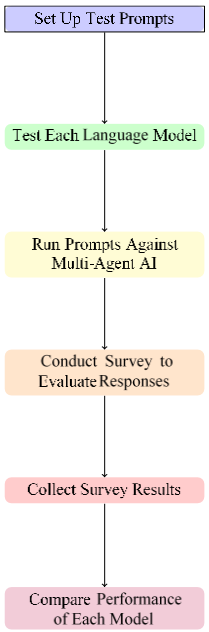

Figure 2. Methodology for Evaluating Multi-Agent AI System

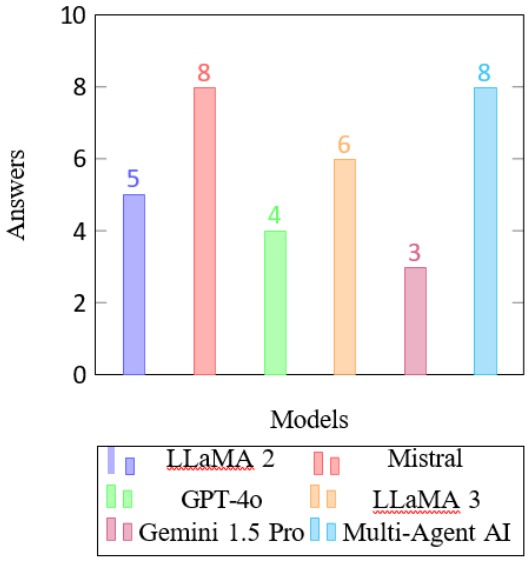

Figure 3. Comparison of LLMs based on answer number across prompts

This study presents a new method of collaborative problem-solving using coordinated multi-agent language models (LLMs). Using the interaction between several LLMs, the proposed system reduces the number of errors, improving understanding of the context, and increasing the overall accu-

racy of AI-based solutions. Unlike traditional approaches based on a single model, this study highlights the benefits of peer review and autonomous error correction, which minimizes the need for human intervention.

The proposed system lays the foundation for future achievements in the field of: developing joint frameworks for multi-LLM collaboration, enabling task-specific specialization among LLMs, and automating error detection and correction in complex problem-solving scenarios.

Table 1. Ey results for each model

Prompt/Language Model	LLaMA 2 13b fp16	Mixtral	gpt4o	LLaMA 3	Gemini 1.5 Pro	Multi-Agent AI
Write a Python script to output numbers from 1 to 100	+	+	+	+	+	+
Write the game «snake» in Python	-	+	+	+	-	+
Write a poem about AI with exactly 50 words	+	+	n/a	n/a	n/a	+
Write an email to my boss letting them know I am leaving the company	+	+	n/a	-	n/a	+
If we lay 5 shirts out in the sun and it takes 4 hours to dry, how long would 20 shirts take to dry? Explain your reasoning step by step	-	+	+	+	+	+
Jane is faster than Joe. Joe is faster than Sam. Is Sam faster than Jane? Explain your reasoning step by step	+	+	n/a	+	n/a	+
There are three killers in a room. Someone enters the room and kills one of them. Nobody leaves the room. How many killers are left in the room? Explain your reasoning step by step	-	+	+	+	+	+
Create JSON for the following: There are 3 people, two males. One is named Mark. Another is named Joe. And a third person, who is a woman, is named Sam. The woman is age 30 and the two men are both 19	+	+	n/a	+	n/a	+

In particular, the development of more autonomous systems is one of the directions of AI development within the frame- work of this research. This work opens up opportunities for practical applications in various fields, from complex decision-making systems to natural language processing via minimizing human control and expanding AI methodologies.

4. Conclusions

Evaluating different language models (LMS) in comparison with a set of tasks demonstrates their advantages and limitations of each of them. Among the tested models, Mixtral has demonstrated reliable capabilities for solving most tasks, which indicates the effectiveness of its expert architecture «several in one». However, this approach has inherent limitations in terms of flexibility. The Mixtral architecture requires the selection of one expert to process each result, which limits parallel collaboration between experts. This needs to be verified in future studies.

In contrast, our multi-agent artificial intelligence system, which includes several films running simultaneously in real-time, used all approaches based on a single model. The ability of the multi-agent system to simultaneously take advantage of the unique advantages of each film provides a higher degree of freedom and productivity in solving a wide range of tasks. This collaboration system can be configured to perform a single repetitive task very well.

The effectiveness of the multi-agent approach to AI highlights its potential to overcome the limitations inherent in individual models. By facilitating real-time interaction and collaboration between different LMS, our system can dynamically adapt to complex and diverse tasks, providing a more reliable and flexible solution than processing data by a single expert in Mixtral.

In general, a multi-agent approach to AI offers a better solution. The simultaneous work of several experts in our system provides higher accuracy and efficiency, which makes it an alternative for real-world applications. This study highlights the need for further development of multi-agent systems that make it possible to use the potential of artificial intelligence and its capabilities beyond existing limitations.

References

[1] Achiam, J. (2024). Gpt-4 technical report. *OpenAI*

[2] Mishra, M., Braham, A., Marsom, C., Chung, B., Griffin, G., Sidnerlikar, D., Sarin, C. & Rajaram, A. (2024). Dataagent: Evaluating large language models' ability to answer zero-shot, natural language queries. *IEEE 3rd International Conference on AI in Cybersecurity (ICAIC)*

[3] Saunders, W., Yeh, C., Wu, J., Bills, S., Ouyang, L., Ward, J. & Leike, J. (2022). Self-critiquing models for assisting human evaluators. https://doi.org/10.48550/arXiv.2206.05802

[4] Cho, W.S., Zhang, P., Zhang, Y., Li, X., Galley, M., Brockett, C., Wang, M. & Gao, J. (2019). Towards coherent and cohesive long-form text generation. https://doi.org/10.48550/arXiv.1811.00511

[5] Kunal, M.R. & Bansal, J. (2023). The future of openai tools: Opportunities and challenges for human-ai collaboration. *2nd International Conference on Futuristic Technologies (INCOFT)*

[6] Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J. & Liu, R. (2020). Plug and play language models: A simple approach to controlled text generation. https://doi.org/10.48550/arXiv.1912.02164

[7] Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Kernion, J., Ndousse, K., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S.,

Olah, C. & Kaplan, J. (2021). A general language assistant as a laboratory for alignment. https://doi.org/10.48550/arXiv.2112.00861
[8] Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B. & Mordatch, I. (2020). Emergent tool use from multi-agent autocurricula. https://doi.org/10.48550/arXiv.1909.07528
[9] Marcus, G. (2018). Deep learning: A critical appraisal. https://doi.org/10.48550/arXiv.1801.00631
[10] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C.L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J. & Lowe, R. (2022). Training language models to follow instructions with human feedback. https://doi.org/10.48550/arXiv.2203.02155
[11] Berman, M. (2024). Matthew berman - youtube channel. *Retrieved from*: https://www.youtube.com/@matthew berman
[12] OpenAI. (2024). Openai leaderboard. *Retrieved from:* https://bit.ly/3qHV0X7

Промттарды өңдеуге арналған өздігінен түзелетін генеративті бағдарлама

К. Уразов*, А. Раззак, Ж. Кальпеева
Satbayev University, Алматы, Қазақстан
**Корреспонденция үшін автор: kaisarkz347@gmail.com*

Аңдатпа. Ірі тілдік модельдер контекстті толық түсінуде қиындықтарға тап болып, тапсырмамен сәйкессіздіктерге жол береді және жиі қателеседі. Бұл адамның араласуын қажет етеді, ал ол өз кезегінде көлемді және уақытты көп талап ететін тапсырмалар үшін тиімсіз. Бұл зерттеу келесі шешімді ұсынады: бірлесіп жұмыс істейтін жасанды интеллект (ЖИ) агенттерінің топтары арқылы тапсырмаларды орындау. Бұл әдісте ЖИ агенттері бір-бірімен қарапайым мәтіндік хабарламалар арқылы ақпарат алмасып, белгіленген мақсатқа жету үшін үйлесімді әрекет етеді. Жүзеге асыру барысында танымал мессенджер API-і қолданылады, ол арқылы ЖИ белгілі бір рөлдерге ие боттар ретінде жасалады. Әр ботқа өзіне тән рөл мен сұраныстар жүктеледі, және олар осы рөлдер мен тапсырмаларға сүйене отырып, ортақ міндетті бірігіп шешеді. Күтілетін нәтижелер: адамның араласуын айтарлықтай азайту, жүйені автоматтандыру деңгейін арттыру, өнімділік пен сенімділікті жоғарылату, сондай-ақ қолдану аясын кеңейту.

Негізгі сөздер: *ірі тілдік модель, көпагенттік LLM ынтымақтастығы, ЖИ негізіндегі жүйелер*.

Самокорректирующееся генеративное программное обеспечение для обработки промтов

К. Уразов*, А. Раззак, Ж. Кальпеева
Satbayev University, Алматы, Казахстан
**Автор для корреспонденции: kaisarkz347@gmail.com*

Аннотация. Крупные языковые модели сталкиваются с трудностями в понимании контекста, часто допускают несоответствия в выполнении задач и совершают ошибки. Это требует частого вмешательства человека, что неэффективно при выполнении масштабных и ресурсоемких задач. В данном исследовании предлагается подход, основанный на взаимодействии команд ИИ и координации их работы для достижения общей цели. Это достигается путем передачи сообщений между ИИ в виде обычного текста. Реализация включает использование API популярного мессенджера, который позволит представить ИИ в виде ботов, наделенных определенными ролями. Каждый такой бот получит свою роль и набор запросов, а, следуя своему назначению и задачам, будет совместно с другими ботами решать поставленную перед группой задачу. Ожидаемые результаты включают значительное снижение необходимости вмешательства человека, повышение уровня автоматизации, улучшение производительности и надежности системы, а также расширение сфер применения.

Ключевые слова: *крупные языковые модели, многоагентное взаимодействие LLM, системы на основе ИИ.*

Received: 20 March 2024
Accepted: 15 September 2024
Available online: 30 September 2024