Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.38118

An Analysis of Zero-Sum Game Agent Training Based on Large Language Models and Self-Play Mechanisms

Silong Liu[1], Chao Shang[1], Yifan Wang[2,*]

[1]*School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China*
[2]*School of Computer Science and Technology, Harbin University of Science and Technology, Harbin, China*
**Corresponding Author. Email: 2304010825@stu.hrbust.edu.cn*

Abstract. This paper carries out a review of competitive agent training under zero-sum game settings, analysing persistent drawbacks found in traditional self-play reinforcement learning, including inflexible decision-making policies and weak long-horizon reasoning capacity. Game agents built purely upon large language models are incapable of sustaining iterative upgrades amid ongoing adversarial contests. Accordingly, the present work investigates viable integration schemes that unite large language models and self-play training systems. It first lays out the definition of zero-sum games, illustrates standard self-play training workflows, and revisits fundamental algorithms from game theory. Four practical integration patterns are classified, namely auxiliary policy generation, reward evaluation optimisation, accumulation of adversarial experience, and the construction of diversified training opponents. Relying on theoretical formula derivation and comparative experimental statistics, this paper evaluates widely adopted modern frameworks against five core benchmarks: reasoning performance, training productivity, generalisation capability, computational expenditure, and training stability, so as to compare their distinct strengths and weaknesses. Evaluation findings indicate that each standalone integration paradigm bears intrinsic flaws. Frameworks adopting dynamically-generated opponents deliver optimal generalisation performance, while optimised reward mechanisms enable the fastest training convergence. Nonetheless, all hybrid training architectures confront common bottlenecks such as LLM hallucination, prohibitive computational overhead, and oscillating gradient signals during model training. This review identifies several high-potential research directions: dynamic multi-module collaborative structures, streamlined self-play workflows, retrieval-augmented technologies to restrain hallucination, as well as the development of general-purpose game agents. The outcomes of this analysis can supply theoretical references for future advances in adversarial decision-making artificial intelligence.

Keywords: Large Language Models, Self-Play, Zero-Sum Games, Game Agents, Reinforcement Learning

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.38118

1. Introduction

Games are important scenarios for testing game strategies and intelligent decision-making. In zero-sum games, the interests of two players are strictly opposed, and their strategies constrain each other, which is effective for evaluating agents' reasoning and adaptability. Current agents mainly achieve iterative strategy improvement through deep reinforcement learning and self-play, but they suffer from certain limitations, including rigid strategies, weak generalization, and insufficient long-horizon reasoning capabilities. Large language models possess strong reasoning and planning capabilities. Agents' decision-making capabilities in complex game environments can be enhanced by integrating them with self-play.

Current research on game agents mainly focuses on three approaches: traditional self-play, game-playing based solely on large language models, and the integration of the two. In traditional research, Zinkevich et al. proposed the Counterfactual Regret Minimization (CFR) algorithm, which solves imperfect-information games by minimizing counterfactual regret [1]. Then, Deep CFR introduced deep neural networks to enhance the ability to handle large-scale games [2]. But there still exist limited high-level reasoning and generalization capabilities in complex scenarios with traditional reinforcement learning methods. Although large language models can assist with game-state analysis and strategic reasoning, they lack a closed-loop training process for continuous optimization. In order to overcome the shortcomings of both methods, research work such as SPIRAL, MARSHAL, and SPIN-LLM is being conducted. They use self-play data to train large language models, bringing game agents to combine reasoning with iterative self-improvement.

This paper will focus on the integration of self-play and large language models. It will first introduce zero-sum game agents, self-play mechanisms, and related reinforcement learning algorithms. Then it focuses on the integration methods of large language models with self-play training in four domains. It also reviews representative studies and compares their performance. Finally, it summarizes the current technical challenges and discusses future development trends. This paper aims to provide a reference for the design of training frameworks and future research on zero-sum game agents.

2. Fundamentals of zero-sum game agents

As early agents depended on pre-designed rules and searches, the capacity of those agents was restricted by the set of rules defined and the size of the search space. With the development of machine learning, the game agents began to use deep neural networks and reinforcement learning algorithms to learn from their experiences in the environment. However, in recent years, with the development of large language models, language understanding and reasoning skills have been leveraged for game state analysis and strategy formation, which has enhanced agents' abilities to generalize and adapt [3]. Self-play mechanisms create training data through self-competition and continuously optimize their strategy based on outcomes. AlphaGo Zero and OpenAI Five are representative examples. AlphaGo Zero starts with a random strategy and improves its abilities of decision making by combining self-play, neural networks, and Monte Carlo Tree Search [4]. OpenAI Five optimizes team strategies in Dota 2 through large-scale self-play and reinforcement learning [5]. Self-play decreases dependence on predefined rules and human-provided data, offering a closed-loop learning paradigm for the ongoing development of agents for zero-sum games.

In the framework of integration of large language models and self-play, traditional algorithms serve as an interface for language-based reasoning and strategy optimization. Monte Carlo Tree Search (MCTS) can expand and select candidate strategies generated by LLMs, Proximal Policy

DOI: 10.54254/2755-2721/2026.38118

Optimization (PPO) can use self-play experience to update LLM decision policies, Counterfactual Regret Minimization (CFR) can guide equilibrium decision-making in imperfect-information environments, and credit assignment mechanisms can transform final game outcomes into finer-grained training feedback.

3. Research on the integration of large language models and self-play training

3.1. The integration of large language models and self-play

The core goal of self-play is to solve for the optimal strategy, that is, to maximize the expectation of discounted cumulative rewards in the whole process of the game under the premise of interacting with the opponent's strategy to generate game trajectories [6]. The large language model has the ability to perform abstract reasoning and open-ended modeling. After being integrated into the closed loop of self-play, it can innovate the traditional self-play system in four aspects: LLM-assisted strategy generation, LLM-assisted reward evaluation, LLM-assisted experience summarization and memory, and LLM-assisted opponent construction [7].

In order to visually describe the complete operation logic of large-model-enhanced self-play, this article summarizes the unified training process, as shown in Figure 1. After the game environment outputs game trajectories, they are sent to four LLM-assisted modules, respectively, completing decision-making action output, intrinsic reward correction, experience buffer purification, and dynamic opponent pool construction. All information is collected to execute policy gradient updates, obtain a new generation of game agents, and feed them back to the game environment, forming a closed-loop iteration.

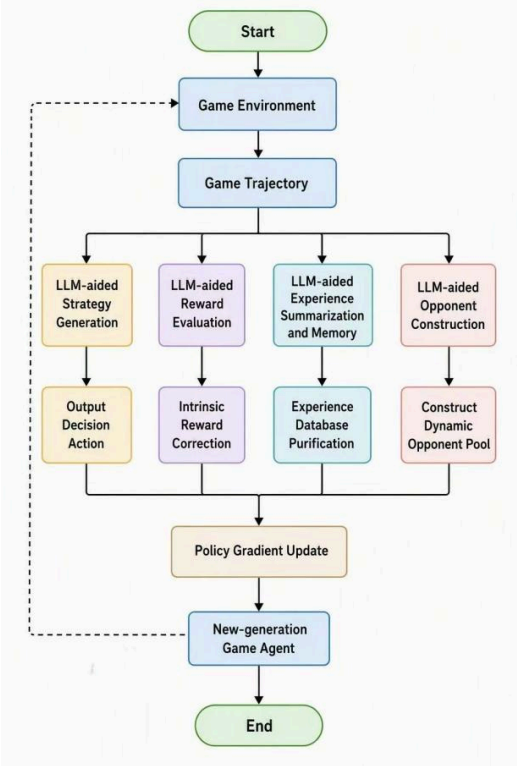


Figure 1. Overall architecture of four fusion paradigms for LLM-enhanced self-play (picture credit: original)

DOI: 10.54254/2755-2721/2026.38118

The above four auxiliary modules intervene in the self-play training process from the dimensions of decision-making output, reward signal, sample processing, and adversarial sample construction, respectively. Each module has different emphases on the improvement of traditional reinforcement learning. In order to clearly compare the performance differences between traditional reinforcement learning and LLM-assisted reinforcement learning in each functional module, a comparative analysis is carried out from the four perspectives of strategy generation, reward evaluation, experience memory and sampling, and adversarial opponent construction, as shown in Table 1.

Table 1. Comparison of improvements in each module of LLM-assisted reinforcement learning under zero-sum games [8-10]

Functional Module	Traditional Reinforcement Learning Methods	LLM-assisted Reinforcement Learning Improvement Methods
Strategy Generation	Performance is bounded by pre-configured action sets, which hinders adaptation to intricate game environments.	Large-language model reasoning is merged with reinforcement-learning policies to remove action-space constraints and support context-aware tactical decisions.
Reward Evaluation	Only terminal rewards are available; intermediate supervisory signals are absent, which slows down training convergence.	LLM-derived intermediate rewards fill gaps in training supervision. They refine parameter updates, speed up convergence, and boost overall model precision.
Experience Memory and Sampling	Excess duplicate samples and uniform sampling behaviour frequently drive the model toward local optima.	LLMs filter out useful experience and adopt diverse sampling schemes to cut redundant training, so the model generalises better across unseen cases.
Adversarial Opponent Construction	Limited opponent diversity creates inadequate training stress, which triggers premature model convergence.	A dynamic opponent bank covering varied behaviours is established. It selects adversarial instances on demand to sustain training pressure and avert premature convergence.

As shown in Table 1, the LLM-assisted reinforcement learning improvement methods have shown significant improvements in four functional modules compared with traditional reinforcement learning, and have gradually become the mainstream zero-sum game self-play assistance method.

3.2. Large language model and self-play integration training framework and typical research

There are three typical LLM and self-play integration training frameworks, namely MARSHAL, SPIRAL, and SPIN-LLM.

MARSHAL focuses on reward evaluation, which is different from the trajectory-level global reward of GRPO. It divides the alternating game into independent round trajectories of different players, introduces round-level fine-grained rewards, and obtains round advantages through cumulative summation and normalization for model update. At the same time, it integrates CFR and Chain of Thought to correct value estimation deviations, realizes the fine-grained attribution of

DOI: 10.54254/2755-2721/2026.38118

game rewards, and improves the generalization ability of the game model. In multi-agent strategy game test scenarios, compared with the traditional PPO baseline, MARSHAL can increase the win rate of game tasks by 28.7% and reduce the game sampling sample size required to achieve Nash equilibrium by about 31%; the fine-grained round reward attribution also reduces training variance, and the stability of model convergence is significantly improved [11].

SPIRAL focuses on strategy generation and is composed of Actor and Learner dual modules. The Actor completes multi-agent self-play sampling to generate game trajectories in parallel through multi-GPU; the Learner introduces role-conditioned advantage estimation (RAE), calculates advantages according to roles and updates the policy network, with weights passed back to the Actor for cyclic iteration, relying on LLM value estimation to alleviate multi-agent training oscillations and improve the model's game reasoning ability. Verification is carried out on the Kuhn Poker zero-sum game task. Relying solely on game self-play training, the model's general mathematical reasoning ability can be improved by 8.7%; compared with the supervised fine-tuning scheme with the same sample size, the reasoning score on the Minerva Math benchmark is improved by 18.1 percentage points, proving that the strategic reasoning acquired in the game process can be transferred to general reasoning tasks [12].

SPIN-LLM focuses on assisting opponent construction and experience assistance. With the Zephyr-7b-sft-full model as the initial base model, the model independently generates synthetic data within each epoch. With the help of the LSPIN loss function, the loss is calculated and backpropagated to update model parameters. The updated parameters flow into the next round of the iteration cycle, and after multiple rounds of iterations, the final fine-tuned lightweight large language model is obtained. The framework constructs strong and weak virtual opponents in layers, builds a dynamic adversarial pool, and improves the generalization ability of lightweight game agents. The experimental results show that compared with the fixed opponent pool training scheme, the dynamic opponent construction strategy increases the model's win rate against unknown opponents by 17.2%; the single-step inference latency of the lightweight model is shortened from 412ms to 276ms, and the peak GPU memory usage is reduced by 2.4GB, balancing both game performance and deployment efficiency [13].

3.3. Comparison and analysis of different integration methods

Based on the above mechanism decomposition of SPIRAL, MARSHAL, and other typical frameworks, this paper systematically compares the characteristics of four types of fusion paradigms from five dimensions such as reasoning ability and training efficiency. The results are shown in Table 2.

Table 2. Multi-dimensional comparison of four major modules of LLM-assisted reinforcement learning

Comparison Dimension	Strategy Generation	Reward Evaluation	Experience Summarization	Opponent Construction
Reasoning Ability	Highest	Medium	Lowest	Second-highest
Training Efficiency	Poor	Highest	Second-highest	Worst
Generalization Ability	Second-highest	Low	Low	Highest
Computational Cost	Highest	Medium	Lowest	Second-highest

DOI: 10.54254/2755-2721/2026.38118

Table 2. (continued)

Training Stability	Poor	Medium	Highest	Worst

The comparison results in Table 2 show that the strategy generation module has outstanding reasoning ability, which can support the high-level logical deduction of complex games, but the computational overhead is large, the training efficiency and stability are poor, and it is difficult to balance high-level reasoning and low-level decision-making. The reward evaluation module has balanced overall performance and high training efficiency, which can effectively alleviate the problem of sparse terminal rewards, but is constrained by LLM scoring hallucinations, and its reasoning and generalization abilities are limited. The experience summarization module has low computational cost and strong training stability, relies on game review to screen high-quality samples, but lacks higher-order reasoning ability. The opponent construction module has the best generalization performance, relying on diversified virtual opponents to avoid the model falling into local equilibrium, but its training efficiency is low, the fluctuation is large, and the inference overhead is high.

The capabilities of the four modules have their own emphasis. There is no single module that can take into account all indicators, and different modules adapt to different game task scenarios. In the face of long-horizon complex game tasks, priority can be given to relying on the strategy generation module to give full play to the reasoning advantages of large language models. Under the condition that computing resources are limited and the model is expected to converge rapidly, reward evaluation and experience summarization can be combined to take into account training efficiency and training stability. If the model needs to adapt to unknown opponents and enhance generalization performance, it is suitable to introduce the opponent construction module to generate a variety of adversarial samples.

4. Technical challenges and future development

4.1. Core technical challenges in current research

Integrating large language models into self-play training pipelines helps overcome certain limitations of traditional reinforcement learning, yet many real-world hurdles persist when researchers try to put this theoretical solution into practice.

Standard self-play systems rely purely on match win-loss results to generate training signals. For multi-agent assignments with lengthy game sequences, this design often sends agent policies into constant fluctuation and undermines overall training stability. The SPIRAL framework applies value-function optimisation and mitigates such unstable behaviour for small-size multi-agent groups. Once the scale of agent teams expands, however, the same stability issues resurface. In imperfect-information games like Texas Hold'em, agents can only capture partial environmental observations, which further amplifies errors during value assessment. MARSHAL pairs CFR algorithms with reasoning modules from large models to adjust flawed value estimates. This method, nevertheless, requires full archival records of every game round; its practical performance falls considerably under real-time online training settings [1, 2]. Researchers have adopted layered reward allocation to resolve sparse-reward challenges in multi-agent games. Even so, splitting total global rewards down to every single move made by agents poses substantial technical barriers and slows model iteration — a recurring problem in dynamic adversarial environments [14].

DOI: 10.54254/2755-2721/2026.38118

Hallucination generated by large language models is another critical factor that disrupts self-play training. Once an agent misreads the current game state, wrong judgments flow through the whole training loop and interfere with gradient updating. After cycles of self-adversarial training, these cumulative errors keep growing larger. Several research projects have adopted knowledge bases to restrict model outputs and reduce hallucination risks, yet such adjustments deliver only modest improvements. In self-play sessions without human-labelled data and explicit game rules, agents frequently pick illegal actions, which lowers the overall quality of training samples [3].

The academic community has developed four mainstream ways to combine LLMs and self-play frameworks, but no universal architecture can adapt itself to all game contexts. Most existing platforms fix the weight value of each functional module without dynamic adjustment throughout different training phases. Under such rigid settings, it becomes tough to strike a fair balance between inference performance and computing resource consumption. Furthermore, the field lacks unified, mature standards for prompt construction. This gap creates major obstacles when researchers transfer well-trained models across distinct game environments [15].

If engineers directly deploy earlier model versions as rival agents in training, adversarial opponents will show repetitive, homogeneous strategies. Agents then have fewer opportunities to explore new tactics and may get stuck in local optima. Though LLMs are capable of generating various simulated opponents and diversifying the opponent pool, this process consumes massive computing resources and yields plenty of low-quality gameplay data. Researchers currently judge opponent quality mostly by win rates, a simple metric that cannot guarantee both sample variety and high sample quality at the same time [16].

Data-related conflicts also deserve close attention. Conventional game agents demand large volumes of manually annotated datasets, which pushes up data acquisition costs greatly. Unsupervised self-play automatically generates game records, yet it frequently produces skewed sample distributions and fails to cover a full range of potential strategies. Large language models can only rework existing training samples instead of fundamentally fixing sample bias. Developers always need to make trade-offs between training performance and computational expenditure [5].

4.2. Future research and development trends

Based on the various real-life bottlenecks identified above, further exploration in this field can be carried out from the perspectives of architecture design, illusion suppression, lightweight training process, and task boundary expansion.

Firstly, a dynamic multi-module collaborative architecture can be studied. Change the current design concept of fixed module parameters and flexibly adjusts the proportion of each module's role according to the different stages of training; gradually establishing a unified paradigm for prompt word design in game scenarios reduces manual parameter tuning workload and enhances the adaptability of the entire framework to different tasks [15].

Secondly, integrating retrieval-based techniques into existing game algorithm systems offers a practical way to suppress the negative impacts brought by model hallucination. In actual training practice, embedding retrieval modules together with self-evolving counterfactual regret algorithms, including Auto CFR, alongside hierarchical reward distribution mechanisms, can effectively correct inaccurate value estimations in environments with incomplete game information. Researchers can further introduce a sample screening mechanism to eliminate irregular and unusable game data. This pre-processing step stabilises the overall data quality and ensures that only valid information is utilised for model updates throughout self-play iterations [14, 17].

DOI: 10.54254/2755-2721/2026.38118

Another feasible optimisation direction lies in streamlining and lightening the overall self-play training procedure. Rather than generating all opponent resources online from scratch, developers can pre-establish a multi-tier opponent library with varying difficulty levels during offline preparation. Only minimal parameter adjustments are required in subsequent online training sessions, which significantly reduces real-time computational loads. In terms of opponent selection strategies, relying merely on win-rate data has proven insufficient. A more reliable alternative involves combining win-rate indicators with strategic entropy metrics as dual evaluation standards. This improved sampling method maintains abundant adversarial diversity while lowering the practical deployment difficulty of intelligent game agents [16].

Finally, it is necessary to optimize the unsupervised sample generation strategy and address the problem of data distribution bias caused by self-play. Breaking out of the research scope of zero-sum games, introducing world models, and transferring the already mature self-game training ideas to more complex real-world tasks such as non-zero-sum games and multi-agent collaborative confrontations, gradually promotes the research and implementation of general adversarial decision-making agents [3, 5].

5. Conclusion

This article provides a systematic review of zero-sum game agent training with large language models combined with the self-play mechanism. First, it reviews the zero-sum game, the fundamentals of self-play training, and the core algorithm of traditional games, analyzes the inherent defects of the lack of generalization of pure reinforcement learning agents and the lack of an iterative closed loop in large language models; it focuses on decomposing the four types of fusion paradigms of LLM-assisted strategy generation, reward evaluation, experience summarization, and opponent construction, and compares the advantages and disadvantages of various schemes in reasoning, training efficiency, generalization ability, computational cost, and stability with experimental data, and reviews the mainstream integration frameworks of SPIRAL, MARSHAL, and SPIN-LLM. At present, there are still core problems in this field, such as LLM hallucination interfering with gradients, fixed fusion module weights, high computational overhead of virtual opponents, and imbalanced sample distribution. Future research will advance toward dynamic multi-module collaborative architectures, hallucination suppression optimization, lightweight training processes, and general cross-scenario game models, further breaking through the technical barriers of large language model logical reasoning and autonomous iteration of self-play, and promoting the implementation of general adversarial decision-making agents.

Author contribution

All the authors contributed equally and their names were listed in alphabetical order.

References

[1] Zinkevich, M., Johanson, M., Bowling, M., et al. (2007). Regret minimization in games with incomplete information. *Advances in Neural Information Processing Systems, 20*, 1729–1736. https://proceedings.neurips.cc/paper/2007/hash/08d98638c6fcd194a4b1e6992063e944-Abstract.html

[2] Brown, N., Lerer, A., Gross, S., et al. (2019). Deep counterfactual regret minimization. *Proceedings of the 36th International Conference on Machine Learning, 97*, 793–802. https://proceedings.mlr.press/v97/brown19b.html

[3] Gallotta, R., Todd, G., Zammit, M., et al. (2024). Large language models and games: A survey and roadmap. *IEEE Transactions on Games, 1*–18. https://doi.org/10.1109/TG.2024.3461510

DOI: 10.54254/2755-2721/2026.38118

[4] Silver, D., Schrittwieser, J., Simonyan, K., et al. (2017). Mastering the game of Go without human knowledge. *Nature, 550*(7676), 354–359. https://doi.org/10.1038/nature24270
[5] Yin, Q. Y., Yang, J., Huang, K. Q., et al. (2023). AI in human-computer gaming: Techniques, challenges and opportunities. *Machine Intelligence Research, 20*, 299–317. https://doi.org/10.1007/s11633-022-1384-6
[6] Silver, D., Hubert, T., Schrittwieser, J., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science, 362*(6419), 1140–1144. https://doi.org/10.1126/science.aar6404
[7] Lanctot, M., Zambaldi, V., Gruslys, A., et al. (2017). A unified game-theoretic approach to multiagent reinforcement learning. *Journal of Artificial Intelligence Research, 60*, 641–692. https://doi.org/10.1613/jair.5512
[8] Silver, D., Huang, A., Maddison, C. J., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature, 529*(7587), 484–489. https://doi.org/10.1038/nature16961
[9] Yu, Y., Yang, Y., & Wang, Y. (2024). Self-play reinforcement learning for multi-agent systems: A survey. *IEEE Transactions on Artificial Intelligence, 5*(3), 389–408. https://doi.org/10.1109/TAI.2024.3372102
[10] Du, Y., Sun, P., Li, J., et al. (2024). Large language model self-play for multi-agent strategic reasoning. In *Proceedings of the 41st International Conference on Machine Learning (Vol. 235)*, pp. 11245–11261. PMLR. https://doi.org/10.48550/pmlr.v235.0672
[11] Yuan, H., Xu, Z., Tan, Z., et al. (2026). MARSHAL: Incentivizing multi-agent reasoning via self-play with strategic LLMs. *Proceedings of the International Conference on Learning Representations*. arXiv. https://doi.org/10.48550/arXiv.2601.08894
[12] Liu, B., Yu, S., Liu, Z., et al. (2026). SPIRAL: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *Proceedings of the International Conference on Learning Representations*. arXiv. https://doi.org/10.48550/arXiv.2506.24119
[13] Chen, Z., Gu, X., & Zhong, L. (2024). SPIN-LLM: Self-play fine-tuning for weak language game agents against strong opponent baselines. *arXiv*. https://doi.org/10.48550/arXiv.2401.01335
[14] Wang, T., Zhao, M. J., Xu, P., et al. (2025). Intelligent decision-making technology for wargaming based on large language models. *Acta Automatica Sinica, 51*(6), 1205–1217. http://www.aas.net.cn
[15] Shan, S. Z., & Zhang, W. W. (2024). Air combat intelligent decision-making method based on self-play deep reinforcement learning. *Acta Aeronautica et Astronautica Sinica, 45*(4), 328723. https://hkxb.buaa.edu.cn
[16] Li, K., Xu, H., Fu, H. B., et al. (2024). Auto-CFR: An automated generation method of counterfactual regret minimization algorithms for imperfect-information games. *Artificial Intelligence, 337*, 104232. https://doi.org/10.1016/j.artint.2024.104232
[17] Yu, L. L., Xu, P. F., Du, H. B., et al. (2026). UAV adversarial game strategy optimization method based on counterfactual regret minimization. *Science China-Information Sciences, 56*(4), 872–888. https://www.springer.com/journal/11432