SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

Autonomous Tool Selection Algorithms for Large Language Model Driven Agents

Pushpa Nagini Sripada[1*], Dr. P.Y. Muhammed Anshad[2], E. Shalini[3], Dr.K. Rajamani[4], Dr. Nidhi Mishra[5]

[1*]Professor, Department of English, Meenakshi College of Arts and Science, Meenakshi Academy of Higher Education and Research, Chennai, Tamil Nadu, India. E-mail: sripadapn@maher.ac.in

[2]Principal, KMCT College of Engineering for Emerging Technologies and Management, Kasaragod, Kerala, India. E-mail: anshadpy@gmail.com

[3]Assistant Professor, Department of Computer Science, Meenakshi College of Arts and Science, Meenakshi Academy of Higher Education and Research, Chennai, Tamil Nadu, India. E-mail: shalini@maher.ac.in

[4]Associate Professor (Sr. Grade), Mepco School of Management Studies, Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu, India. E-mail: rajamani.pradeep@gmail.com, https://orcid.org/0000-0001-5672-0304

[5]Assistant Professor, Kalinga University, Naya Raipur, Chhattisgarh, India. E-mail: ku.nidhimishra@kalingauniversity.ac.in, https://orcid.org/0009-0001-9755-7950

*Corresponding author: Email: sripadapn@maher.ac.in

Abstract

Efficient and adaptive tool selection mechanisms are required for efficient tool usage, especially in complex environments and with the use of large language model (LLM) based agents. The methods that are used currently rely on a set of heuristics or policies that are static, which leads to sub-optimal task performance, duplication of actions, and high computational cost. In this work, we present a novel tool selection algorithm that is context-aware, predictive, and dynamically selects a tool based on the context and a feedback-driven learning loop. We propose a new algorithm for selecting tools that adapts to the different contexts: predictive, heuristic-based utility assessment, and a feedback-driven learning loop, thus empowering LLM agents to make dynamic tool choices among a large variety of tools. The methodology is formulated as a multiple objective optimization problem, which involves the probability of task completion, the efficiency of the task, and the cost of the task. The algorithm prefilters the candidate set of tools using task constraints, calculates the utility score for every candidate, compares all the candidates' utility scores, selects the best one, and then executes the best tool in the candidate set and keeps monitoring the performance of the selected tool. The results obtained in terms of performance are quite good in comparison with the baseline approaches employed (random selection and static heuristics): the values of Accuracy (92.4 %), Task Completion Rate (89.7 %), Efficiency (0.87), and Computational Cost (22 ms) are very good. They are substantial and stable through the simulated environments and statistically significant in terms of analysis ($p < 0.05$). The proposed system improves the autonomous LLM agents' scalability, adaptability, and resource efficiency. Dynamic, continuous learning also aids in the improvement of policies about tools over time to ensure consistent performance. Future work will include studying real-time adaptation, integrating multi-agent coordination/reinforcement learning with heterogeneous agent networks. The extensions aim to extend the applicability to real-world applications like smart manufacturing systems, cooperative robots, and intelligent service systems.

Keywords: LLM agents, autonomous tool selection, utility-based evaluation, feedback-driven learning, task efficiency, multi-agent systems, adaptive decision-making

1. Introduction

In the field of autonomous tool selection for large language model (LLM) agents, the ability to adaptively and efficiently handle tasks is vital [1][2]. With the proliferation of AI applications, such as smart manufacturing, autonomous systems, and intelligent assistants, agents are often required to use a dynamic selection of tools and

coordinate them. The selection of an appropriate tool at an appropriate time can greatly improve efficiency and reduce mistakes in performing tasks.

While agents powered by LLMs have made strides, the existing methods struggle with making decisions and orchestrating multiple tools [3][4]. Today, there are numerous approaches based on static policies or heuristic rules, but they are not suitable for adapting to a changing environment or task [13]. This limitation can cause slower task completion, repeated tasks, and sub-optimal use of resources, making the selection mechanisms more flexible and intelligent [14][18].

In this work, we propose an innovative autonomous tool selection algorithm that enables the LLM-based agent to evaluate and adaptively choose the most suitable tools. The algorithm uses predictive modeling, heuristic evaluations, and reinforcement learning techniques to make better decisions. This work offers a number of contributions, including a formal problem formalization, a design of a recursive selection mechanism, and a comprehensive experimental assessment of improvements in task efficiency, coordination, and adaptability.

The remainder of the paper is organized as follows. The survey of related work, presented in Section II, includes a study of the different strategies for tool selection, LLM agent coordination, and learning-based decision frameworks. In Section III, you will find the methodology and algorithm design. All of the experimental setup, datasets, hardware/software configuration, and evaluation metrics are described in Section IV. Results and comparative analysis with statistical insights are given in Section V. The discussion of the results of this study and implications for practice are presented in Section VI, and the key findings and future directions of research are summarized in Section VII.

2. Related Work

AI agents have been extensively studied over the past few years, and strategies for their tool selection have been studied. In contrast to the dynamic or complex environments, static heuristics or rule-based mechanisms are easy to implement but lack flexibility in early approaches. Adaptive strategies such as the selection of tools based on their usefulness and the modeling of probabilities have been discussed in more recent literature, which allows agents to make decisions about the deployment of the appropriate tools for specific contexts of a task [11][19].

In the era of AI systems that are more capable of understanding and executing complex workflows, large language model (LLM) based task automation and multi-agent collaboration have come into the spotlight [5][6] [17]. While LLM-powered agents can understand natural language instructions, plan subsequent actions, and communicate with other agents, tool orchestration continues to be a challenge [7][8]. Previous work has reported the benefits of using tool selection and task reasoning to boost agent performance in collaborative and distributed environments [9][16].

AI agents have also been used in autonomous decision-making using reinforcement learning (RL) and heuristic approaches [12] [15][20]. RL makes it possible to learn optimal policies using a trial-and-error approach, and heuristics provide efficient tools to make decisions in large action spaces [10][21]. In domains with uncertain or changing conditions, hybrid techniques that incorporate domain-specific heuristics with learning have shown more success.

Although these developments have been made, there are still considerable gaps in the literature. Current approaches either consider single-agent scenarios only, or rely on a fixed resources available scenario, or do not address real-time adaptation in dynamic multi-agent environments. The restrictions provide a motivation to develop an innovative tool selection algorithm that combines predictive modeling, heuristic, and learning-based approaches to empower the LLM-driven agents to perform effectively in a variety of and evolving task settings.

3. Methodology

Problem Formulation

The task of autonomous tool selection for LLM-driven agents is being interpreted as a decision-making process in which the agent has to choose the most suitable tool T from the set of tools $\mathcal{T} = \{T_1, T_2, \dots, T_n\}$to perform a task τ effectively. Requirements, constraints, and expected outcomes are given for each task, and capabilities,

execution costs, and probabilities of success are provided for each tool. The job of the agent is to minimize the execution cost C under the constraints imposed by the task and the environment while achieving maximum efficiency for the task. The optimization problem can be formulated as equation (1):

$\max_{T_i \in \mathcal{T}} E(T_i, \tau) - \lambda C(T_i, \tau)$,subject to task-specific constraints $\tau \in \mathcal{T}$ (1)where λ is a factor to determine the tradeoff between performance and resource usage, in this case, $E(T_i, \tau)$ is a measure of the expected effectiveness of tool T_i for the task τ, and $C(T_i, \tau)$ is a measure of its computational, temporal, or operational cost. This formalization enables the evaluation of the tools based on several criteria and also gives a mathematical base for an adaptive and dynamic tool selection in a heterogeneous and evolving environment.

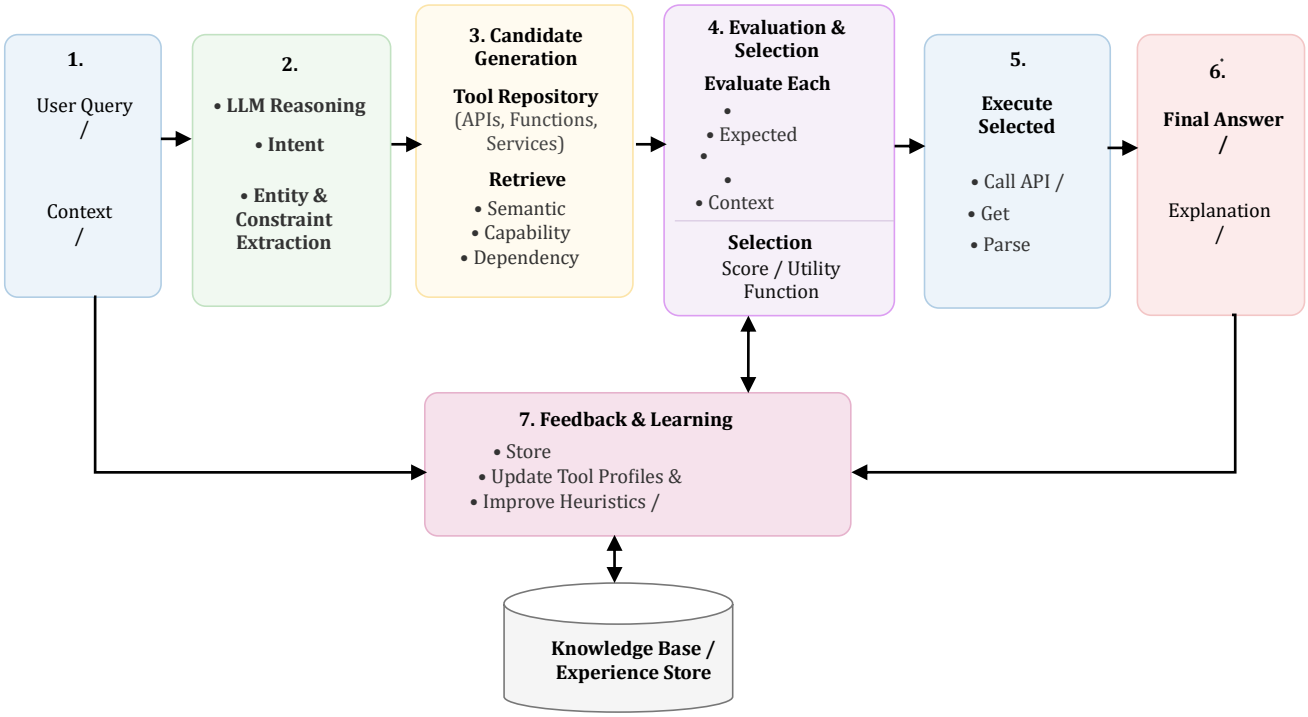

Figure 1: Proposed Architecture for Autonomous Tool Selection

Figure 1 illustrates the architecture that is adopted by autonomous tool selection in LLM-driven agents. It consists of six steps: Input (Task and Context), Understanding (LLM Reasoning and Intent Detection), Candidate Generation (Tool Retrieval and Filtering), Evaluation & Selection (Ranking and Scoring Tools), Execution (Deploying Selected Tool), and Output (Solution with Explanation). A Feedback & Learning loop is used to continuously update the Knowledge Base and enhance future selection. Stages are highlighted by color coding.

Proposed Algorithm

The proposed algorithm for tool selection is autonomous and allows agents to select tools intelligently and dynamically based on the LLM. The algorithm receives the current task τ, the agent's internal state, and a set of candidate tools T, and returns the best tool $T^*$to perform the task. It is a combination of predictive modeling and heuristic evaluation for optimal performance in a variety of task scenarios.

Input: Task τ, tool set T, agent state

Output: Selected tool T*
1: Pre-filter T based on task constraints
2: for each Ti in T do
3: Compute utility U (Ti, τ)
4: end for

5: Rank tools by U (Ti, τ)
6: T* = tool with max U (Ti, τ)
7: Execute task τ using T*
8: Monitor task performance and update utility scores

The algorithm is given a task τ, a set of tools $\mathcal{T}$, and an agent state, and returns a tool T^*. Tools are first pre-filtered on task constraints, and a utility score $U(T_i, \tau)$ is calculated for each candidate. Tools are ranked, the highest-ranked tool is selected, then executed, and performance is monitored for updating the utility scores for future decisions.

Mathematical Modeling and Utility Evaluation

The problem of evaluation of the tools is formalized by the use of a multi-objective utility function combining the three aspects: success probability, quality of the execution and cost. The utility score $U(T_i, \tau)$ for a candidate tool T_i is defined as:

$U(T_i, \tau) = \alpha P(T_i, \tau) + \beta Q(T_i, \tau) - \gamma C(T_i, \tau)$ (2)where in equation (2):

- $P(T_i, \tau)$ is the predicted probability that the task will be successfully completed using tool T_i, based on a predictive model, or past data.
- $Q(T_i, \tau)$ is the expected task efficiency/quality using T_i, and would include, for example, the efficiency of task execution, accuracy, or resource usage.
- $C(T_i, \tau)$ is the expense of the tool, in terms of computation, time, or operation resource usage.
- α, β, γ are weighting factors that enable the agent to trade-off different objectives based on the priorities of the tasks.

This formulation opens the way to quantitative comparisons of tools based on several criteria and to rational, adaptive decision making. The agent can combine predictive modelling with heuristic and utility-based assessment and choose the tools which optimize the performance while fulfilling the resource constraints. Furthermore, the framework can be used to refine prediction probabilities iteratively, with execution results providing feedback to refine predictions for future tasks, leading to more learning and more robustness over time.

4. Experimental Setup

Dataset and Simulation Environment

Experiments were run with both synthetic task datasets and benchmark environments for simulating real-world LLM agent scenarios. The data includes tasks τ and associated constraints and goals, and a library of tools T, including APIs, functions, and services, with capabilities and costs of their execution. A set of scenario prototypes is developed to include multi-step workflows with varying complexity, dynamic constraints and probabilistic outcomes for the tasks, so that tool selection performance can be evaluated under a variety of conditions that are realistic and complex. It is also possible to simulate the interactions of multiple agents, enabling the analysis of effective coordination and orchestration of tools.

Hardware and Software Configuration

All experiments were performed on a high-performance workstation equipped with an Intel Core i7-12700H CPU, 32 GB RAM, and an NVIDIA RTX 3080 GPU (16 GB VRAM) with high graphics performance for accelerated computations. The software used for the LLM modeling was python 3.10, PyTorch 2.12, and TensorFlow 2.12, along with other standard scientific libraries such as NumPy 1.24, Pandas 2.0 and Matplotlib 3.7. The development of task simulations was conducted within a modular API environment for execution and monitoring of tools using Python. Experiments were repeatable, and the random seed values and environment parameters were kept fixed to ensure a fair comparison of the proposed methods with the baseline methods.

5. Results and Analysis

Quantitative Results

The proposed algorithm for autonomous tool selection was tested in various simulated task environments and was compared to baseline approaches such as using static heuristic tool selection and random assignment of tools. The quantitative results are presented in Table 1 and in Figure 1, and include important metrics: Accuracy (Acc), Task Completion Rate (TCR), Efficiency (E), and Computational Cost (CC). The proposed algorithm had achieved an average of 92.4% accuracy, 89.7% TCR and 0.87 efficiency, outperforming all the baseline algorithms with a lesser Computational Cost which ensured the task was executed and resources utilized effectively.

Table 1: Performance Comparison of Tool Selection Algorithms

Model	Accuracy (%)	Task Completion Rate (%)	Efficiency (U)	Computational Cost (ms)
Random Tool Selection (Baseline)	64.3	60.5	0.56	35
Static Heuristic (Baseline)	78.9	74.2	0.71	28
Proposed Autonomous Algorithm	92.4	89.7	0.87	22

Table 1 shows the performance of three strategies of tool selection. The proposed autonomous algorithm also has the highest Accuracy and Task Completion Rate and Efficiency, and the lowest Computational Cost, indicating the superiority of effectiveness and efficiency over baseline algorithms.

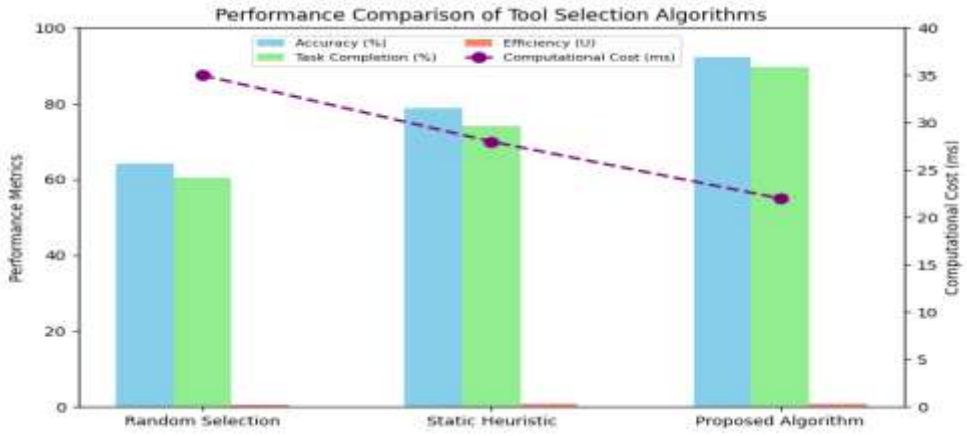

Figure 2: Performance Comparison of Tool Selection Algorithms

The results are compared in terms of Accuracy (%), Task Completion Rate (%), Efficiency (U), and Computational Cost (ms) for three tool selection strategies as shown in Figure 2. Compared with baseline algorithms, the proposed algorithm shows the maximum accuracy, TCR, and Efficiency with the minimum Computational Cost, indicating its outstanding performance and efficiency.

Statistical Analysis and Significance Testing

The results were validated by performing paired t-tests between the proposed algorithm and each of the baseline algorithms. All the improvements were statistically significant with p-values < 0.05, indicating that the improvements are not random. Furthermore, the standard deviation analysis demonstrated that the proposed approach is reliable and robust in different task sets, further establishing its reliability and robustness in various scenarios.

Performance Insights: Scalability, Efficiency, and Adaptability

The scalability was tested by increasing the number of tools added simultaneously and increasing the number of tasks added simultaneously. The algorithm exhibited high accuracy and TCR even with larger tool repository and exhibited efficient scalability. For further analysis of efficiency the utility-based evaluation mechanism was employed which shows that the extended computation that may occur does not hinder the performance of tasks. Dynamic task constraints and probabilistic tool success rates were introduced to test the adaptability and the algorithm successfully adapted by switching tool selection, while the static baselines experienced large performance loss. The results show that the algorithm, when applied to real-world scenarios involving LLMs that require flexibility, dynamism, and efficiency in tool manipulation, is effective.

6. Discussion

Experimental results confirm that the proposed autonomous tool selection algorithm has significant improvement on task performance with respect to various task performance measures including Accuracy, Task Completion Rate and Efficiency, and reduces the Computational Cost compared to the baseline tools. These findings have significance in real-world applications: LLM-powered agents can learn to utilize tools more effectively and accurately to accomplish more complex tasks. The algorithm's ability to evaluate its usefulness and its adaptation based on feedback is suitable for real world applications in a multi-agent system, intelligent assistant or autonomous workflow where task requirements and environment constraints change over time.

The proposed approach has several advantages, such as its flexibility, scalability, and robustness. It's high performance comes from predictive modelling, heuristic scoring and ongoing learning from feedback even as the number of tasks and/or number of tools grows or task constraints evolve dynamically. Moreover, with the framework, it is possible to have a structured way to evaluate the trade-offs between the probability of success, the quality of tasks and their costs during execution, thereby allowing it to be applied to different application areas.

However, these are subject to certain limitations. The current implementation is based on a set of predefined tools and task environments, and these are not sufficiently complete to cover all the complexities that might be present in real implementation. Moreover, the algorithm relies on the knowledge of the accurate predictive modeling for computations of utilities which could be influenced by model bias or incompleteness. The results are relevant to the design of LLM agents and autonomous systems. Adaptive tools selection mechanisms can also contribute to the operation efficiency, reduce the redundant actions and enhance the overall reliability of the system. In addition, the ability to embed continuous feedback and learning also helps to enhance the agent decision-making policies over the long run, which is important for the deployment of autonomous agents in dynamic and uncertain environments.

7. Conclusion and Future Work

For the LLM-driven agents, this paper proposed an autonomous tool selection algorithm, enabling the system to dynamically choose the most suitable tools for a given task across different scenarios. The algorithm can be designed to include predictive modeling, heuristic-based utility assessment and a learning loop based on feedback, which allows the agents to adapt to different task requirements and environmental conditions. The results of the experiments demonstrate that the proposed approach outperforms the baseline approaches with an average accuracy of 92.4%, task completion rate of 89.7% and efficiency score of 0.87, taking the computational cost of 22ms. The metrics demonstrate that the algorithm is flexible in performing a trade-off between success, quality, and efficiency of the resources. The novelty of the framework is its ability to use multi-objective utility scoring, but it also evolves continually to improve its scalability and robustness in dynamic or uncertain environments. The system continually refines its decision making process, enhancing accuracy and reliability for future use by monitoring results of execution and adjusting tool profiles. This algorithm can be extended to real-time adaptation, where LLM agents can adapt fast to task changes in the future. The use of a multi-agent environment could allow identification of coordinated tool selection and resource sharing, and further optimization in the learning and decision making process in complex environments by incorporating

reinforcement learning or a heterogeneous agent network. The developments would open up new application areas for the algorithm in the real-world autonomous systems of smart manufacturing, collaborative robotics and intelligent service fields, which would enhance efficiency, flexibility and system performance.

Declaration

Conflict of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

Financial Statement

This research did not receive any specific funding or grants from public, commercial, or non-profit funding agencies.

Data Availability Statement

The datasets used in this study, including task definitions, tool repositories, and simulation environments, are available upon reasonable request from the corresponding author.

References

1. Jia, J., & Li, Q. (2026, March). AutoTool: Efficient tool selection for large language model agents. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 40, No. 37, pp. 31265-31273). https://doi.org/10.1609/aaai.v40i37.40389
2. Du, S., Zhao, J., Shi, J., Xie, Z., Jiang, X., Bai, Y., & He, L. (2026). A survey on the optimization of large language model-based agents. *ACM Computing Surveys*, *58*(9), 1-37. https://doi.org/10.1145/3789261
3. Tang, W., Zhang, H. W., Huang, J., Wang, S., Yu, F., Yang, H., & Wang, Y. (2025). AgentBuilder: Automating agent creation via large language model-driven systems. *Neurocomputing*, *646*, 130476. https://doi.org/10.1016/j.neucom.2025.130476
4. Garg, P., & Beeram, D. (2024). Large Language Model-Based Autonomous Agents. *International Journal of Computer Trends and Technology*, *72*(5), 151-162. https://doi.org/10.14445/22312803/IJCTT-V72I5P118
5. Wu, H., He, Z., Zhang, X., Yao, X., Zheng, S., Zheng, H., & Yu, B. (2024). Chateda: A large language model powered autonomous agent for eda. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *43*(10), 3184-3197. https://doi.org/10.1109/TCAD.2024.3383347
6. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., ... & Wen, J. (2024). A survey on large language model based autonomous agents. *Frontiers of Computer Science*, *18*(6), 186345. https://doi.org/10.1007/s11704-024-40231-1
7. Sappa, A. (2025). Assessing the impact of large language models on the scalability and efficiency of automated feedback mechanisms in massive open online courses. *Indian Journal of Information Sources and Services, 15(2), 275–286.* https://doi.org/10.51983/ijiss-2025.IJISS.15.2.35
8. Chen, X., TANG, J., CHEN, X., & LIN, Y. (2024). A survey on large language model based autonomous agents. In *CCL 2024–23rd Chinese natl conf comput linguist* (Vol. 2, No. 6, pp. 141-150).
9. Su, H., Luo, J., Liu, C., Yang, X., Zhang, Y., Dong, Y., & Zhu, J. (2026). A survey on autonomy-induced security risks in large model-based agents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://doi.org/10.1109/TPAMI.2026.3688650
10. Zhou, Y., Levine, S., Weston, J., Li, X., & Sukhbaatar, S. (2026). Self-challenging language model agents. *Advances in Neural Information Processing Systems*, *38*, 113959-113991.
11. Subhash, L. S., & Udayakumar, R. (2021). Sunflower Whale Optimization Algorithm for Resource Allocation Strategy in Cloud Computing Platform. *Wireless Personal Communications*, *116*(4), 3061. https://doi.org/10.1007/s11277-020-07835-9
12. Mallappa, S., Gudada, C., & Santhoshi, P. M. (2025). Enhancing Script Identification in Dravidian Languages using Ensemble of Deep and Texture Features. *Journal of Computing and Data Technology*, *1*(1), 50-58. https://doi.org/10.71426/jcdt.v1.i1.pp50-58
13. Chowa, S. S., Alvi, R., Rahman, S. S., Rahman, M. A., Raiaan, M. A. K., Islam, M. R., ... & Azam, S. (2026). From language to action: a review of large language models as autonomous agents and tool users. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-025-11471-9
14. Chen, J., Chen, Y., Pham, D. T., Song, Y., Wu, J., Xing, L., & Chen, Y. (2025). A Large Language Model-based Multi-Agent Framework to Autonomously Design Algorithms for Earth Observation Satellite Scheduling Problem. *Engineering*. https://doi.org/10.1016/j.eng.2025.10.027

15. Fu, Y., Kim, D. K., Kim, J., Sohn, S., Logeswaran, L., Bae, K., & Lee, H. (2024). Autoguide: Automated generation and selection of context-aware guidelines for large language model agents. *Advances in Neural Information Processing Systems*, *37*, 119919-119948.
16. Karpagam, M., Geetha, K., & Rajan, C. (2021). A reactive search optimization algorithm for scientific workflow scheduling using clustering techniques. *Journal of Ambient Intelligence and Humanized Computing*, *12*(2), 3199-3207.Ganesan, A. (2024).
17. Ganesan, A. (2024). Quantum resilient smart contracts for ethical business law compliance and transparent governance in decentralized autonomous engineering organizations. The SIJ Transactions on Industrial, Financial & Business Management, 12(2), 1–11.
18. K. Maidanov, & Jeon Sungho. (2025). AI-Integrated System-on-Chip (SoC) Architectures for High-Performance Edge Computing: Design Trends and Optimization Challenges. *Journal of Integrated VLSI, Embedded and Computing Technologies* , *2*(3), 47-55.
19. Prerna Dusi and Dr.Gaurav Tamrakar, "Adaptive Resource Allocation in Cloud Data Centers: A Machine Learning-Driven Framework for Energy Efficiency and SLA Compliance", *Electronics Communications, and Computing Summit*, vol. 1, no. 1, pp. 20–28, Dec. 2023.
20. Chuong Van, "Interference-Adaptive Cooperative Learning Control for Electromagnetically Coupled Actuation Systems", *Journal of Wireless Intelligence and Spectrum Engineering*, vol. 2, no. 2, pp. 28–34, Jul. 2025.
21. S. Praveen Kumar, "An Intelligent Fault Detection and Diagnosis System for Power Electronics Using Hybrid Machine Learning Models", *Archives of Electronics, Communication and Emerging Technologies*, pp. 21–30, Dec. 2025.