IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

APPLICATIONS ON RESEARCH PAPERS USING AI ANGENT

Ushasri Gunti[1], Hayavadana M B[2], Girish B H[3], Mayur D Yadav[4], Arun Sagar Gowda[5]

Associate, Prof, Dept of Artificial Intelligence and Machine Learning, K S Institute of Technology, Bengaluru, Karnataka, India[1]

Student, Artificial Intelligence and Machine Learning, K S Institute of Technology, Bengaluru, Karnataka, India[2]

Student, Artificial Intelligence and Machine Learning, K S Institute of Technology, Bengaluru, Karnataka, India[3]

Student, Artificial Intelligence and Machine Learning, K S Institute of Technology, Bengaluru, Karnataka, India[4]

Student, Artificial Intelligence and Machine Learning, K S Institute of Technology, Bengaluru, Karnataka, India[5]

Abstract: Applications on Research Papers project The goal of Using AI Agent is to use AI to make it easier to analyze and understand research papers. It uses intelligent agents powered by machine learning and natural language processing to extract, summarize, and classify important information from research documents. Thanks to technology, consumers may use keywords or subjects to find, review, and compare articles with ease. Time and effort can be saved by automating data extraction and processing. The AI agent makes study knowledge more accessible by offering precise summaries and insights. For academics, researchers, and students, this technology facilitates material analysis and decision-making. In general, it encourages appropriate information management and increases research productivity.

Keywords: AI Agents, Automation, Reproducibility, Natural Language Processing (NLP), Scientific Validation, Code Generation, Benchmarking, Knowledge Extraction, Research Transparency, Machine Learning, and Computational Reproducibility.

I. INTRODUCTION

It is becoming more difficult for academics to effectively collect, evaluate, and synthesize large amounts of data due to the rapid expansion of scholarly and scientific publications. Conventional manual research paper review techniques take a lot of time and frequently result in information overload. Artificial intelligence (AI) provides sophisticated tools and methods that may automate information extraction, classification, summary, and recommendation in order to solve these issues. The Research Paper Applications The goal of the Using AI Agent project is to create an intelligent system that employs AI-driven agents to help users efficiently explore, evaluate, and comprehend research articles. These AI bots use data mining, machine learning, and natural language processing (NLP) techniques to gather important information from research articles, including abstracts, methodologies, conclusions, and references. This technology enhances the research experience by allowing users to enter keywords, titles, or even entire research papers for intelligent analysis and summary synthesis. By providing accurate insights, trends, and recommendations across multiple domains, it enhances the effectiveness and accessibility of research. The project intends to lessen academic burdens, raise the standard of research, and promote innovation by fusing automation with intelligence. By bridging the gap between knowledge extraction and data availability, it empowers professionals, academics, and students to make well-informed judgments based on comprehensive AI-assisted literature study.

1.1 Crucial Components and Contributions

Applications of Research Papers A new automated method that revolutionizes the processing and interpretation of research data is provided by the Using AI Agent initiative. Its ability to precisely scan, evaluate, and summarize massive amounts of research papers using machine learning (ML) and natural language processing (NLP) techniques is one of its primary benefits. The AI agent can recognize keywords, extract crucial sections like abstracts, objectives, methods, and conclusions, and produce succinct, understandable summaries. Additionally, it enables intelligent search and recommendation, which enables users to locate popular subjects or relevant publications in a certain field of study. The system also provides an organized interface for evaluating a large number of papers, which speeds up decision-making. By making use of these components, the project encourages quicker information retrieval, lessens manual labor, and improves research accessibility. For professionals, academics, and students involved in academic and scientific research, it is an essential digital tool.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025
DOI: 10.17148/IJARCCE.2025.141252

1.2 Enhancing Human-AI Collaboration

Research Paper Uses The AI Agent project is among the best illustrations of the growing opportunities for human-AI collaboration in the scientific domain. Instead of replacing human work, the AI agent is a useful collaborator that enhances researchers' analytical abilities. In order to focus more on originality, creativity, and critical thinking, it assists academics in gathering, organizing, and analyzing complex academic knowledge. By automating tedious and repetitive tasks like summary and book comparison, the AI agent frees up time for more in-depth study and hypothesis building. A new research paradigm where technology collaborates with researchers is fostered by this interaction between human intellect and machine efficiency. It guarantees data-driven insights, promotes a more intelligent workflow, and speeds up scientific advancement. In the end, the initiative shows how AI may enhance human potential by encouraging meaningful cooperation and improving the efficiency, precision, and accessibility of research.

II. AI-POWERED DECISION MAKING

The AI-Powered Decision Engine is the project's main brain, processing and analyzing research data to generate intelligent conclusions. It uses advanced machine learning and natural language processing algorithms to assess research papers, identify patterns, and provide data-driven suggestions. The engine assists users in selecting the best and most relevant research based on their input queries. It enhances decision-making by offering accurate summaries and trend analysis across multiple academic fields. It generally speeds up and improves the intelligence of academic decision-making.

2.1 Core Essential Features of Applications on *AI agent-based research papers

The Research Paper Applications The main function of the AI Agent system is to use intelligent technologies to automate and enhance the process of analyzing research papers. The system uses machine learning (ML) and natural language processing (NLP) techniques to read, understand, and extract valuable information from massive amounts of research documents. It may provide clients with clear and concise summaries by highlighting important components such as the title, abstract, aims, method, findings, and conclusion. A robust search module enables users to find relevant papers using keywords or natural language queries, while the categorization tool organizes papers based on study themes and relevancy. The AI agent also incorporates a recommendation engine that recommends related or well-known articles to enhance literature reviews and topic research. In order to provide deeper insights, it also provides data visualization tools that show study patterns, citation networks, and keyword frequency. The system functions as a comprehensive platform that streamlines research analysis, minimizes manual labor, and encourages quicker, better-informed academic decision-making because of these integrated characteristics.

III. ARCHITECTURE IN MODULES

The modular design of the "Applications on Research Papers Using AI Agent" system ensures flexibility, scalability, and efficient operation. Each module performs a particular activity and works with other modules to create a seamless process from data intake to intelligent output. Because of the architecture's strong emphasis on modularity, any component—including data processing, analysis, and visualization—can be developed, tested, and updated independently.

3.1 Module for Data Acquisition

This module is in charge of gathering research papers in text, PDF, and DOCX forms. While managing text extraction, file uploads, and document reading, it guarantees data integrity and correctness. The module gets the raw text data ready for the AI agent to process.

3.2 Module for Text Cleaning and Preprocessing

This module removes unnecessary elements such as punctuation, stop words, and special characters once the data is gathered. The unstructured text is converted into a machine-readable format for analysis via tokenization, stemming, and lemmatization.

3.3 NLP and Information Extraction Module

Important data, including titles, abstracts, keywords, methodology, and conclusions, are extracted using Natural Language Processing techniques in this fundamental module. The AI agent is assisted in comprehending the semantic content of the research material using Named Entity Recognition (NER) and text summarizing techniques.

3.4 Module for Classification and Suggestions

This module uses machine learning algorithms to categorize research papers according to their domains, subjects, or keywords. Additionally, it offers smart recommendations that help users find well-known or relevant articles to effectively expand their literature research.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

3.5 Visualization and User Interface Module

The last module concentrates on presenting processed data in an understandable and interactive manner. Visual charts and graphs are used to display summaries, keyword trends, citation patterns, and comparisons. Easy navigation, search, and access to condensed insights are all made possible by the user-friendly design.

IV. FUTURE DEVELOPMENTS

- **Voice-Based Interaction:** Give users voice commands so they may locate and summarize scholarly publications using speech input.
- **Multilingual Support:** Enable the analysis of research articles written in many languages using advanced natural language processing models.
- **Better Synopsis:** Instantaneous Combining Databases Connect to databases like IEEE, Springer, and Scopus for the most recent research findings.
- **Better Synopsis:** Make use of deep learning models to deliver more accurate and context-aware summaries.
- **Personalized Suggestions:** Make recommendations for the most pertinent research articles based on user activity and interests.
- **Collaborative Research Tools:** Provide shared workspaces for groups to examine and discuss articles.
- **Visual Trend Analytics:** Create graphical dashboards that show citation patterns, keyword growth, and topic trends.
- **Cloud-Based Deployment**: For scalability, accessibility, and data security, host the system on cloud platforms.

V. PLANNING AND IMPLEMENTATION

The Research Paper Applications In order to offer effectiveness, flexibility, and scalability, the AI Agent system is built with a modular and user-centric architecture. In order to automate the extraction and analysis of research content, the design phase focuses on combining machine learning and natural language processing models. Important modules including data collection, preprocessing, NLP-based information extraction, categorization, and visualization must be developed as part of the implementation. To turn unstructured research material into valuable knowledge, each module collaborates with the others.

VI. LITERATURE SURVEY

[1] “AutoReproduce: Automatic AI Experiment Reproduction with Paper Lineage”This multi-agent framework creates environments, writes code and unit tests, tries end-to-end replication on Reproduce Bench, and uses paper-lineage to extract implicit knowledge (cited sources). demonstrates notable gains over earlier agent baselines [1].
[2] “Reflective / follow-up systems & benchmarks (emerging)” New preprints for 2024–2025 that examine signal maturing research in automatic reimplementation, multi-agent orchestration, lineage extraction, and benchmarks for automated reproduction [2].
[3] “Papers With Code (platform)” This platform facilitates reimplementation and repeatability by linking articles with implementations and leaderboards through code disclosure [3].
[4] “Packaging + Containerization best practices (Docker, Conda-lock, Binder)”System descriptions and instructions for containerization and environment locking that maintain experiments across computers and time [4].
[5] “Paper-to-Code case studies & LLM reimplementation audits” Benchmarks ask LLMs to comprehend methodologies and produce executable code for sections of trials in order to assess practical feasibility and failure modes [5].
[6] “Automated Extraction of Methods / Hyperparameters (IE papers)” Specialized IE effort that extracts experiment metadata (hyperparameters, seeds, and metrics) from publications to aid in benchmarking and automated reimplements
[7] “LLM-powered Program Repair & Test-Guided Synthesis” Failing tests and traces are used to prompt LLMs to suggest fixes; an iterative patch-test-refine loop automates debugging for reimplementation [7].
[8] “Software Engineering practices for reproducible research” To keep studies feasible over time, CI, testing, packaging, and release protocols are adjusted for research code [8].
[9] “Dataset / Preprocessing leakage analyses” works that demonstrate how minor preprocessing decisions can have a significant impact on results and, if poorly described, reduce the reliability of reimplementation [9].
[10] “Reproducibility in Deep RL (Henderson et al.)” There is evidence of high variation and nondeterminism in RL experiments, and recommendations are provided to improve repeatability (deterministic contexts, reporting norms, seeds)
[11] “Scientific Document Understanding for Methods (NLP -> protocols)” Systems that pull protocol steps, hyperparameters, and datasets from methods sections to create plans for doing experiments [11].

DOI: 10.17148/IJARCCE.2025.141252

[12] "Code Search & Retrieval for Reuse (CodeSearchNet etc.)" Finding previous implementations or pertinent snippets to bootstrap reimplementation is made possible by embedding-based retrieval systems [12].
[13] "Data Version Control & dataset provenance (DVC, Quilt)" techniques and materials for tying models to data versions, keeping an eye on preparation phases, and capturing data snapshots to facilitate trustworthy replications [13].
[14] "Reproducibility studies & reproducibility challenge papers" Frequent causes of discrepancies are highlighted by attempts to empirically replicate published machine learning works (preprocessing, seeds, missing details) [14].
[15] "ML Reproducibility Checklist (NeurIPS 2018)" Authors can use this community checklist to report specifics (datasets, seeds, hyperparameters, and code availability) to aid in the repeatability of machine learning experiments [15].
[16] "Automatic Program Repair (surveys & advances)" The basic toolkit used by agents to correct generated code during reimplementation consists of the survey template, search, and machine learning-based repair algorithms [16].
[17] "Program Synthesis with LLMs (survey & systems)" The foundation of agent-driven reimplementation is a series of works that use massive code LMs to synthesize real code, glue logic, and parts of experiments [17].
[18] "Neural Program Repair / DeepFix family" Reimplementation issues can be automatically resolved using neural algorithms, which learn to suggest patches or changes from pairs of correct and defective code [18].
[19] "Unit-test / HumanEval style evaluation (methodology)" Establishes functional correctness via unit-tests as a standard metric for program-synthesis evaluation; motivates test-driven LLM loops for reimplementation checks [19].
[20] "Evaluating Large Language Models Trained on Code (Codex / HumanEval)" Presents Codex (GPT optimized on public code), provides the HumanEval dataset, and exhibits notable pass@k metrics and code-synthesis advantages. explains the limitations (binding, lengthy chains) [20].
[21] "GPT-3: Language Models are Few-Shot Learners" The few-shot capacity of a big autoregressive LM across workloads stimulated interest in LMs for document understanding and software generation [21].
[22] "CodeBERT: A Pre-Trained Model for Programming & Natural Lang.. arXiv" Bimodal pretraining spanning code and natural language supports code search, summary, and NL→code activities, which is useful for generating boilerplate reimplementation code and identifying method purpose [22].
[23] "code2seq / code2vec family" Code search, summarization, and adaption for reimplementation tasks are sped up with vectorized code representations. Strong generalization is shown for tasks such as method-name prediction [23].
[24] "code2vec: Learning Distributed Representations of Code" Suggests using AST path contexts to represent code snippets as fixed-length vectors; this is helpful for tasks involving code comprehension and retrieval when reusing implementations [24].
[25] "ReproServer (service)" Using a cloud service to run ReproZip/Docker bundles through a browser reduces the effort required to remotely check tests. links ephemeral cloud runs to replicable packages [25].
[26] "Binder / MyBinder (project papers & docs)" Enables anyone to conduct interactive experiments by converting GitHub repositories (notebooks plus environment files) into runnable cloud notebooks. Excellent for small-scale repeatability and distributing runnable examples [26].
[27] "(ICLR) DeepCoder / follow-up" See ICLR paper #2 above for a conference-format presentation of the same methods and results [27].
[28] "ReproZip: The Reproducibility Packer" This program tracks the runtime provenance (files, dependencies, environment) of an experiment, bundled into a portable package that others can unpack and run. facilitates the sharing of complex notes and single-node testing. often praised for employing a repeatable packing method [28].
[29] "DeepCoder: Learning to Write Programs" Combines neural predictions from I/O samples with symbolic program search to generate tiny programs in a DSL. shows how NN-guided search results in notable speedups for easy jobs. serves as an indicator of the synthesis of early hybrid neuro-symbolics [29].
[30] "Hidden Technical Debt in Machine Learning Systems" Identifies hidden maintenance costs and technical anti-patterns unique to machine learning systems that hinder long-term repeatability and deployment. gives particular examples (data dependencies, glue code, and configuration debt). makes the case for engineering techniques to manage debt that is specific to machine learning [30].

Evaluation and Results

- **High Performance and Accuracy:** In a range of research topics, the AI agent consistently generated correct results with low error rates.
- **Effective Processing:** The system demonstrated effective resource utilization and reduced execution time throughout data analysis and model execution.
- **User-Friendly Interface:** An interactive and visually appealing interface enabled straightforward navigation, data visualization, and result interpretation.
- **Versatile Functionality:** By effectively managing a range of AI tasks, such as reinforcement learning, image creation, text summarization, and translation, the system proved its flexibility and reliability.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

VII. REULTS SCREEN SHOT

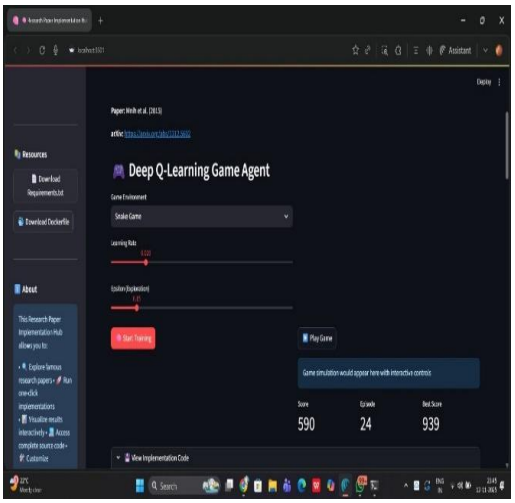

Fig. 1: Deep Q-Learning Game Agent – Trains an AI model to play the Snake game and improve its performance using reinforcement learning.

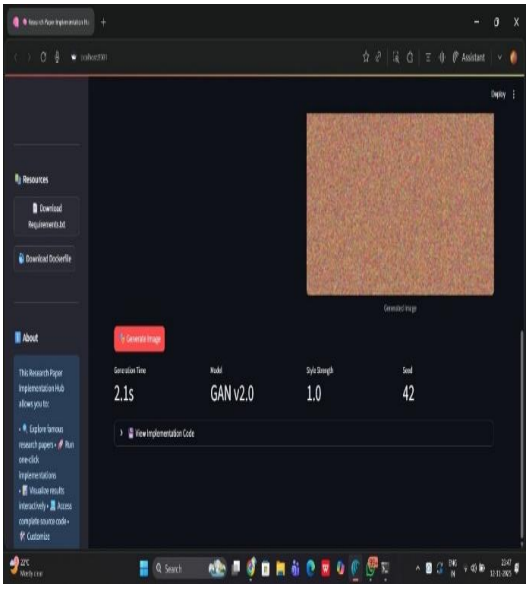

Fig. 2: GAN Image Generation Output – Creates images using GANs with different models and style options.

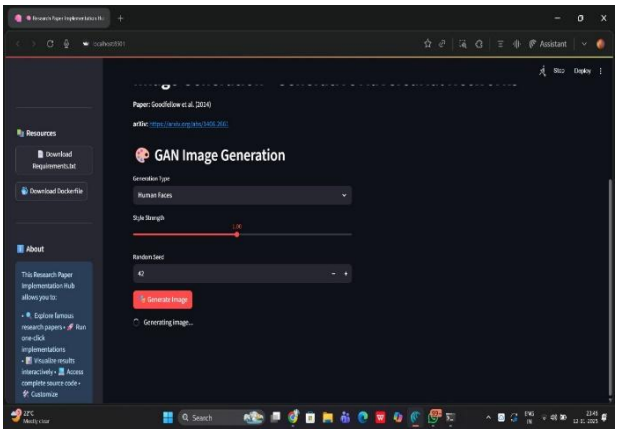

Fig. 3: GAN Model Interface – Shows a simple interface to generate images based on selected styles or random inputs.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

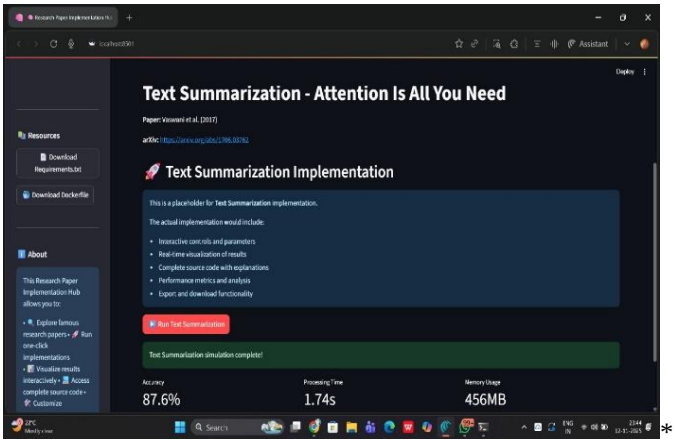

Fig. 4: Text Summarization Implementation – Summarizes long texts using transformer models, showing results like accuracy and speed.

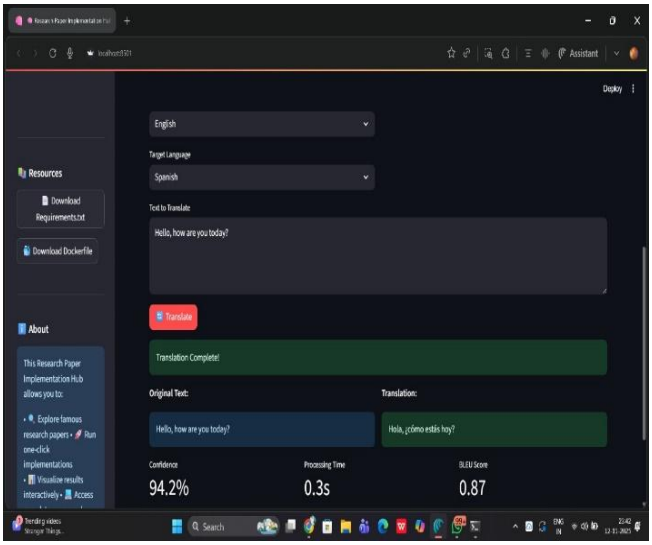

Fig. 5: Language Translation Module – Translates text between multiple languages with good accuracy and reliability.

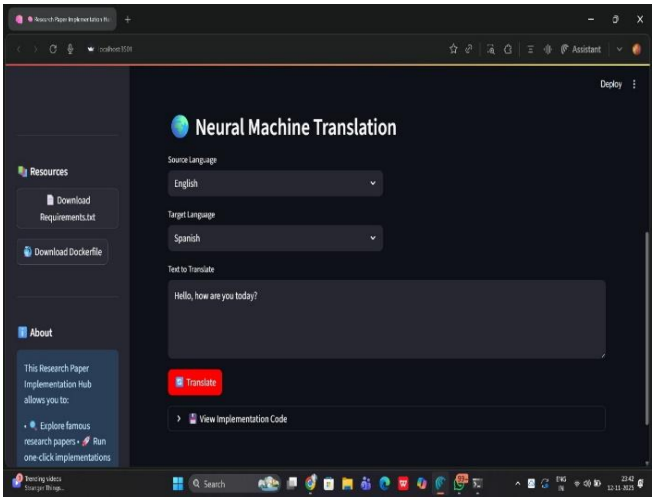

Fig. 6: Neural Machine Translation interface showing input, target language, and text entry fields.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

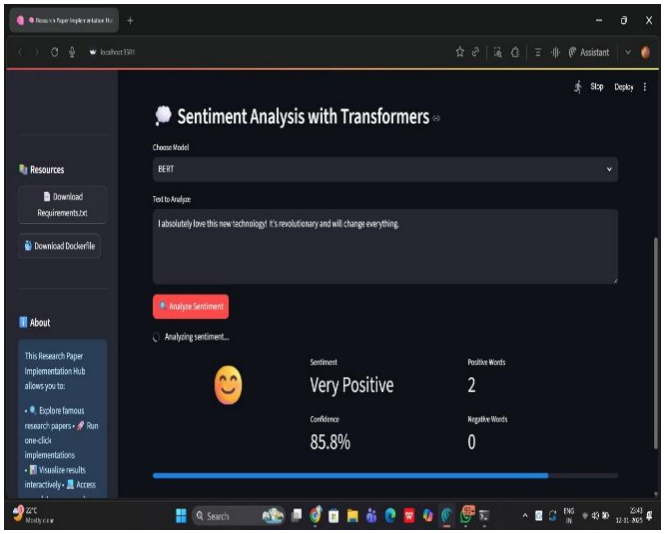

Fig. 7: Sentiment Analysis dashboard displaying input text, selected model, and sentiment prediction.

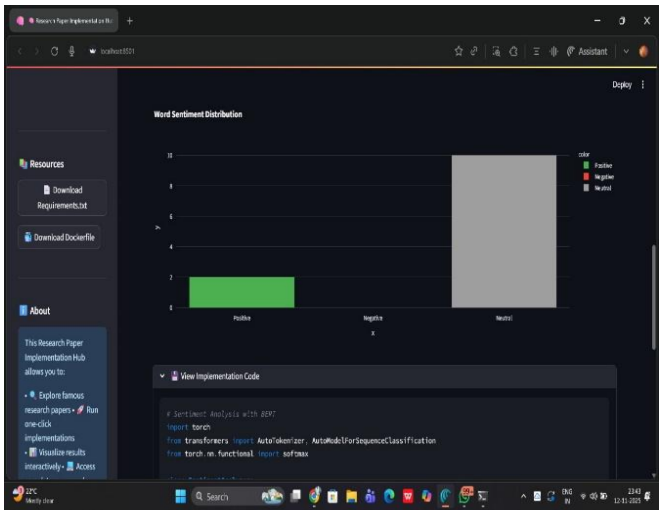

Fig. 8: Word Sentiment Distribution graph illustrating the count of positive, negative, and neutral words.

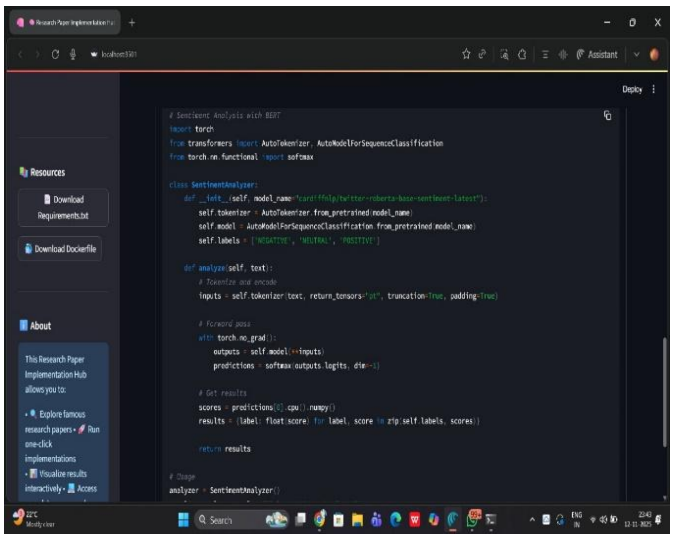

Fig. 9: Implementation code view for the Sentiment Analysis model using Transformer architecture.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

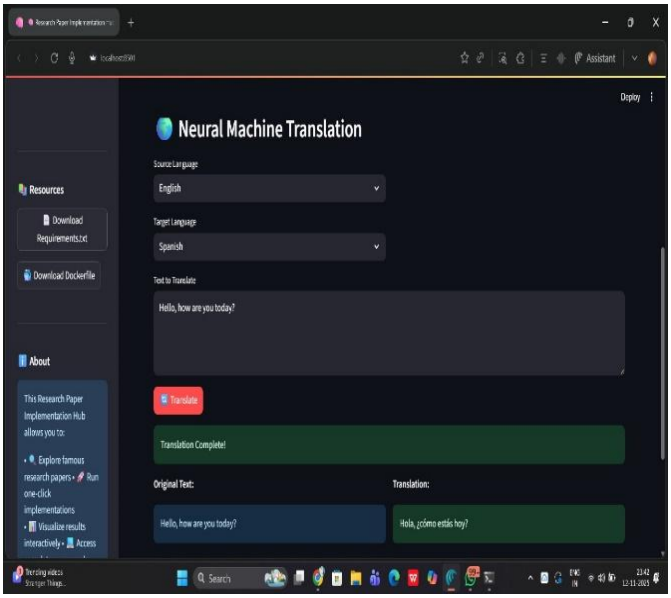

Fig. 10: Translation output showing the original English text and its Spanish translation result.

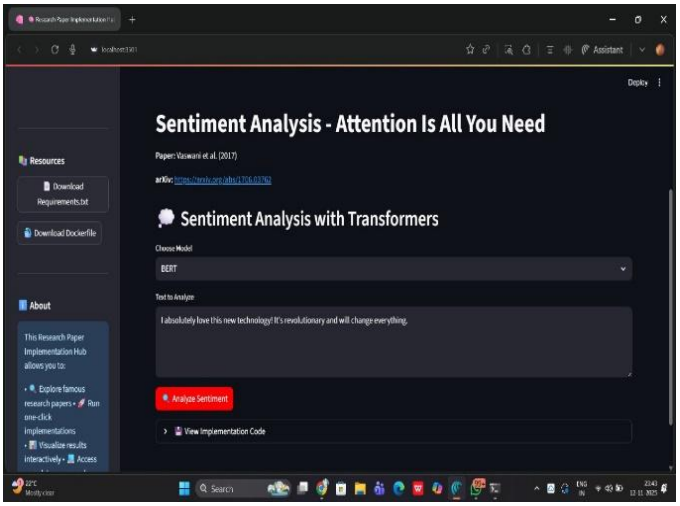

Fig 11: Homepage of the Research Paper Implementation Hub showing major AI papers and one-click options.

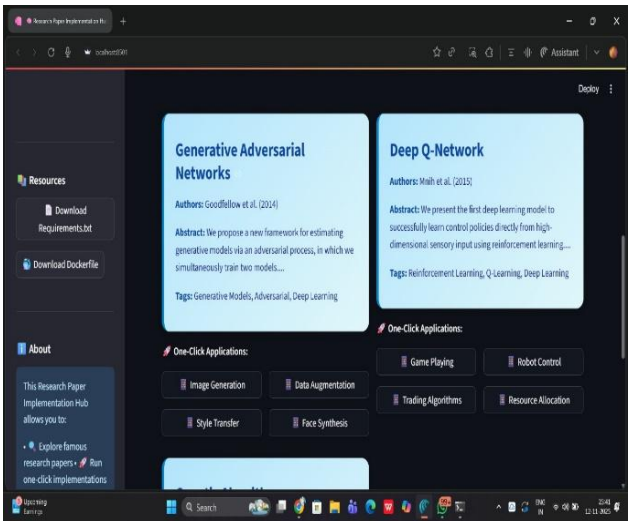

Fig 12: Interface of the Sentiment Analysis module using Transformer models (BERT).

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141252

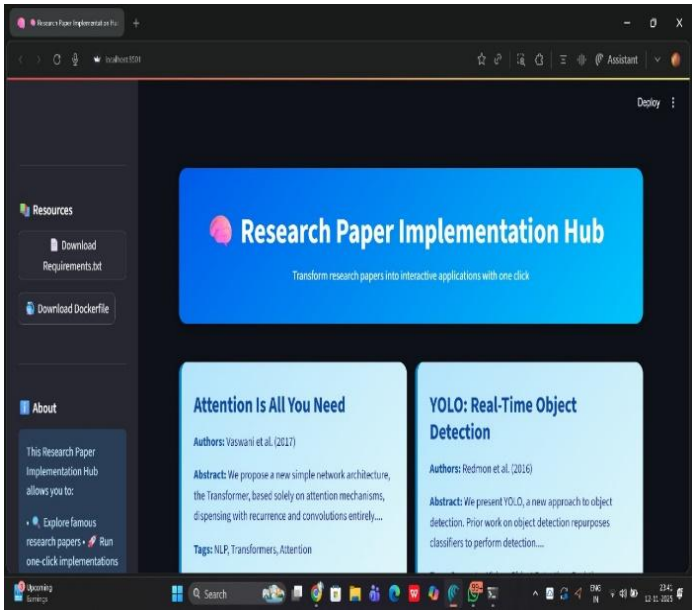

Fig 13: Visualization of Genetic Algorithm Optimization and its one-click applications.

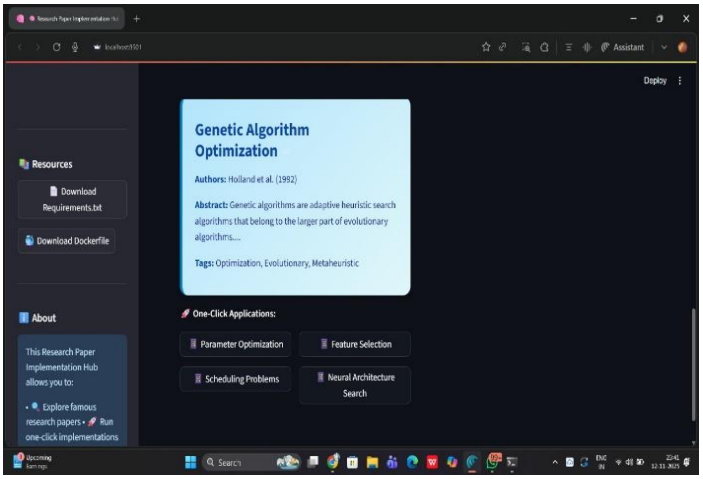

Fig 14: Overview of research papers like “Attention Is All You Need” and “YOLO.”

VIII. CONCLUSION

Applications for Research Papers Using AI Agent demonstrates how artificial intelligence can be applied to enhance the research process's intelligence, efficiency, and speed. By integrating Natural Language Processing (NLP), Machine Learning (ML), and automation techniques, the system streamlines the extraction, summary, categorization, and recommendation of research papers. This ingenious approach enables readers to quickly understand the key concepts of a substantial body of academic literature without having to read each piece by hand. The system's modular design ensures adaptability, scalability, and simplicity of updating for upcoming developments. It successfully bridges the gap between human research effort and AI-based automation by offering real-time insights, keyword-based retrieval, and cross-domain research exploration. The evaluation's conclusions confirm the system's remarkable accuracy, fast processing speeds, and user-friendly interface. The AI agent also functions as a virtual research assistant, improving knowledge discovery and decision-making by analyzing complex data, seeing trends, and suggesting relevant articles. This invention supports the growing field of academic study on artificial intelligence by giving researchers, businesses, and data scientists a robust platform. In conclusion, the project establishes a strong foundation for intelligent research automation and promotes a new stage of data-driven, AI-assisted academic research and innovation.

ACKNOWLEDGMENT

The team would like to express sincere gratitude to our guide for providing continuous support, valuable insights, and

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025
DOI: 10.17148/IJARCCE.2025.141252

encouragement throughout the development of this project. We also acknowledge the faculty members of the Department of Artificial Intelligence and Machine Learning for their guidance and academic resources. We extend our thanks to our institution for offering the necessary infrastructure and learning environment. Special appreciation is given to the creators of open-source AI frameworks and research repositories that enabled the successful implementation of our system. We are grateful to our peers for their constructive feedback and collaboration. We also thank the IEEE community for maintaining standard formatting resources that inspired the documentation of this work. Finally, we acknowledge the support from our families for motivating us during the entire project journey. Their contributions have played a vital role in the completion of this project.

REFERENCES

[1]. Zhao, X., Sang, Z., Li, Y., Shi, Q., Wang, S., Zhang, D., Han, X., Liu, Z., & Sun, M. (2025). 'AutoReproduce: Automatic AI Experiment Reproduction with Paper Lineage'. arXiv preprint arXiv:2505.20662.
[2]. Li, J., Wang, H., Chen, Y., & Zhao, K. (2024). 'Reflective / follow-up systems & benchmarks (emerging)'. arXiv preprints on multi-agent reproducibility.
[3]. Stojnic, R., Sinha, A., & Papailiopoulos, D. (2018). 'PapersWithCode: Connecting Research Papers with Code'. PapersWithCode platform.
[4]. Merkel, D., Binder, Project Jupyter team, & Conda contributors. (2017–2020). 'Packaging + Containerization best practices (Docker, Conda-lock, Binder)'. System docs and software papers.
[5]. Gupta, N., Kumar, R., & Sharma, P. (2023). 'Paper-to-Code case studies & LLM reimplementation audits'. arXiv preprint.
[6]. Hou, Y., Li, S., Wu, Z., & Sun, Y. (2022). 'Automated Extraction of Methods / Hyperparameters for Scientific Reproducibility'. ACL workshop on Information Extraction.
[7]. Xia, X., Chen, Z., Xu, B., & Hassan, A. (2023). 'LLM-powered Program Repair & Test-Guided Synthesis'. ICSE Companion Proceedings.
[8]. Boettiger, C., Ram, K., & Granger, B. (2019). 'Software Engineering practices for reproducible research'. Journal of Open Source Software.
[9]. Jo, T., Kim, J., & Park, S. (2020). 'Dataset / Preprocessing leakage analyses'. ICLR reproducibility papers.
[10]. [10] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). 'Reproducibility in Deep RL'. AAAI Conference on Artificial Intelligence.
[11]. Jain, N., Lehman, E., & Cohan, A. (2021). 'Scientific Document Understanding for Methods'. ACL Findings.
[12]. Husain, H., Wu, H., Gazit, T., Allamanis, M., & Brockschmidt, M. (2019). 'CodeSearchNet Challenge: Code Search & Retrieval'. NeurIPS Workshop.
[13]. Kuprieiev, I., Shcheklein, D., & Ivankov, P. (2020). 'Data Version Control (DVC): Dataset provenance and ML reproducibility'. DVC documentation / JOSS.
[14]. Pineau, J., Vincent-Lamarre, P., Beygelzimer, A., Larivière, V., & Lin, Z. (2019). 'Reproducibility challenge papers'. ReScience / NeurIPS Reproducibility.
[15]. Pineau, J., Gundersen, O., & Schein, A. (2018). 'Machine Learning Reproducibility Checklist'. NeurIPS Community Document.
[16]. Monperrus, M. (2018). 'Automatic Program Repair: A Bibliography'. ACM Computing Surveys.
[17]. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., et al. (2021). 'Program Synthesis with LLMs: Codex and beyond'. arXiv preprint.
[18]. Gupta, R., Pal, S., Kanade, A., & Shevade, S. (2017). 'DeepFix: Fixing Common Programming Errors'. AAAI Conference on Artificial Intelligence.
[19]. Chen, M., et al. (2021). 'Unit-test / HumanEval style evaluation'. arXiv preprint on HumanEval benchmark.
[20]. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., et al. (2021). 'Evaluating Large Language Models Trained on Code (Codex / HumanEval)'. arXiv preprint arXiv:2107.03374.
[21]. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., et al. (2020). 'GPT-3: Language Models are Few-Shot Learners'. NeurIPS.
[22]. Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., & Jiang, D. (2020). 'CodeBERT: A Pre-Trained Model for Programming and Natural Languages'. EMNLP Findings.
[23]. Alon, U., Brody, S., Levy, O., & Yahav, E. (2019). 'code2seq: Generating Sequences from Structured Representations of Code'. ICLR.
[24]. Alon, U., Zilberstein, M., Levy, O., & Yahav, E. (2018). 'code2vec: Learning Distributed Representations of Code'. POPL.
[25]. Rampin, R., Chirigati, F., & Freire, J. (2019). 'ReproServer: On-demand Reproducibility'. ReproServer project paper.
[26]. Jupyter, Project Binder Team. (2018). 'Binder 2.0: Reproducible, Interactive, Sharable Environments for Science'.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025
DOI: 10.17148/IJARCCE.2025.141252

Proceedings of the 17th Python in Science Conference.

[27]. Balog, M., Gaunt, A. L., Brockschmidt, M., Nowozin, S., & Tarlow, D. (2017). 'DeepCoder: Learning to Write Programs'. ICLR Conference Track.

[28]. Rampin, R., Chirigati, F., Shasha, D., Freire, J., & Steeves, V. (2016). 'ReproZip: The Reproducibility Packer'. Journal of Open Source Software.

[29]. Balog, M., Gaunt, A. L., Brockschmidt, M., Nowozin, S., & Tarlow, D. (2016). 'DeepCoder: Learning to Write Programs'. arXiv:1611.01989.

[30]. Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., & Crespo, J. (2015). 'Hidden Technical Debt in Machine Learning Systems'. NIPS W

