OPEN ACCESS
International Journal of Advances in Artificial Intelligence and Machine Learning Vol 3 No 1 (2026) https://doi.org/10.58723/ijaaiml.v3i1.457
Check for updates
Research Article

Pythinsearch: A Simple Web Search Engine

Received: May 29, 2025	Revised: July 08, 2025	Accepted: March 02, 2026	Publish: March 30, 2026

Annam Rupa *, Sadhu Swathi Priya, G.Sumana, M. Navya Sri, N.Chandana

Abstract:

Background: The rapid growth of web content has increased the complexity of retrieving relevant and high-quality information, especially in resource-constrained environments. Traditional keyword-based search engines often fail to capture semantic relationships and structural importance within web documents, leading to suboptimal retrieval performance.

Aims: This study aims to develop a lightweight and modular web search engine, PyThinSearch, that integrates content-based and link-based ranking techniques to improve retrieval effectiveness and efficiency in low-resource and domain-specific environments.

Method: The proposed system employs a hybrid ranking approach combining TF-IDF, PageRank, and HITS algorithms, along with anchor text analysis to enhance contextual relevance. The system is designed using a modular pipeline architecture consisting of data crawling, text preprocessing, indexing with inverted index, ranking, and query processing. Performance is evaluated using standard information retrieval metrics, including Precision, Recall, F1-score, Mean Average Precision (MAP), Normalized Discounted Cumulative Gain (NDCG), and response time.

Result: The experimental results demonstrate that the hybrid ranking model consistently outperforms individual methods. The system achieves higher retrieval effectiveness, with improvements in Precision (0.78), Recall (0.75), MAP (0.77), and NDCG (0.80). Additionally, anchor text analysis significantly enhances performance in ambiguous queries, while the inverted index structure ensures efficient query response times suitable for small- to medium-scale datasets.

Conclusion: PyThinSearch provides an effective and efficient solution for information retrieval by integrating textual relevance and structural importance within a lightweight and modular framework. The proposed system is well-suited for deployment in resource-constrained environments, although future work should focus on incorporating advanced NLP techniques and scalable architectures to improve performance in large-scale applications.

Keywords: Applied Machine Learning; Anchor Text Analysis; Information Retrieval; Lightweight AI Systems; HITS Algorithm.

1. INTRODUCTION

In the modern digital era, the internet has become a fundamental platform for information access, communication, and knowledge sharing (Yaqub & Al-Sabban, 2023). With the exponential growth of web content, users are increasingly challenged to retrieve relevant, accurate, and high-quality information from vast and continuously expanding data sources. Search engines play a critical role in addressing this challenge by enabling efficient navigation and retrieval of information within complex digital environments and contextual relevance between documents (Xiong et al., 2024). As a result, users may encounter redundant, irrelevant, or low-quality results, especially in domains that require high information accuracy, such as cybersecurity, artificial intelligence, and technical knowledge repositories (Bragilovski et al., 2025; Lyu et al., 2025).

Traditional search engines predominantly rely on keyword-based matching techniques to retrieve documents (Kayest & Jain, 2022; Nadim et al., 2023). While these methods are effective for identifying content containing specific query terms, they often fail to capture deeper semantic relationships (von Hippel & Kaulartz, 2021).

In addition to semantic limitations, conventional search systems often overlook the structural importance of web documents within the broader network. The

interconnectivity of web pages through hyperlinks and anchor texts provides valuable information about document authority and relevance (Ajjam & Al-Raweshidy, 2026; Breit et al., 2023).

While ranking models have improved, keyword retrieval models, in comparison, are limited in identifying and using structural context relationships, such as quality links, to be the most authoritative on a specific topic, making it almost impossible to improve the search results (Guo et al., 2022).

While there have been notable improvements in large-scale search engine technologies, there is still a notable dearth of resource-efficient, lightweight, and modular search engine technologies optimized for resource-constrained systems. Most existing solutions necessitate larger infrastructures with high computational costs, making them ill-suited for educational purposes, small-scale, and customized domain-specific applications (Kadyrbek et al., 2025).

This study proposes the modular and lightweight PyThinSearch, a web search engine built in Python. The engine synthesizes both content and link algorithms to rank search results. This study employs TF-IDF, alongside PageRank and HITS. In addition, anchor text analysis is also integrated to improve contextual understanding and, in turn, result ranking.

While most search engine prototypes tend to be function-oriented, PyThinSearch is also a contextual system oriented in its computational efficiency, and both its design and performance contribute to user effectiveness. The system's enhancement and modularity lend themselves to its flexibility and efficiency in its deployment in resource-constrained systems.

This study contributes:

1. The development of a lightweight web search engine framework integrating content-based and link-based ranking algorithms.
2. The incorporation of anchor text analysis to enhance contextual relevance in search results.
3. An exhaustive system-side and retrieval effectiveness evaluation.
4. A practical system design that supports deployment in resource-constrained environments, such as educational platforms and small-scale applications.

This research is among the early works that showcase the use of efficient, flexible, and lightweight design in modular search systems that integrate both content and structural analysis for deployment in resource-constrained search systems in the context of artificial intelligence.

This study differs from previous studies that tackle the design of large-scale search infrastructures, offering a lightweight systems design that can balance search capabilities and practical design constraints.

2. MATERIAL AND METHOD

Data Acquisition and Crawling Module

Collecting web pages from the internet is the role of the Data Acquisition Module. Data is acquired largely as a result of two methods (Booij et al., 2022; Vijayan et al., 2021). First, many Python libraries are available for manual web crawling, such as Requests and BeautifulSoup. Second, one can make use of openly available datasets, like Wikipedia and the TREC collections.

In this work, the lightweight and integrative web search engine framework, which draws on both content-based and link-based information retrieval frameworks, is referred to as PyThinSearch. This system differs from most of the conventional experimental approaches in that it follows a systems approach, where all the components of the system are designed as a single retrieval pipeline (Fan et al., 2022). The framework has been designed with scalability and efficiency in mind to ensure that it can be deployed easily in environments with limited and constrained resources (Pandey et al., 2025; Santos et al., 2021).

The modular system of PyThinSearch is seen in Figure 1 along with the interaction of the different components of the system.

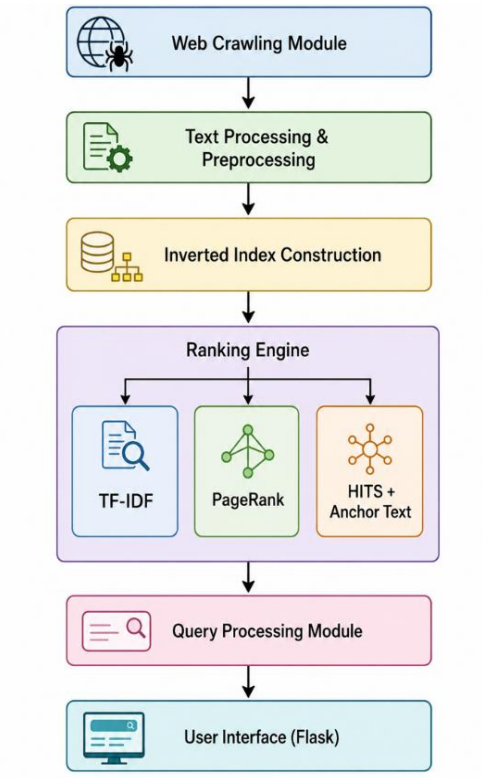

Figure 1. PyThinSearch System Architecture

As depicted in Figure 1, the system operates a pipeline that encompasses all the structured steps from data collection to user request fulfillment (Göppert et al., 2021). Each of the system components within the framework can be optimized independently due to the modular architecture. The integration of content-based and link-based ranking within the Ranking Engine is an important design that improves retrieval accuracy.

There are four main components within the framework of the system: the Data Acquisition and Crawling Component, the Text Processing and Indexing Component, the Ranking and Retrieval Engine, and the User Query Interface (Amir Mehmood & Tahir, 2024; Bifulco et al., 2021). Components are designed to work automatically in a sequence to accomplish the task of retrieving and

Text data on the web is collected by crawling pages and extracting the text itself, the hyperlinks, and the data that is used to create the hyperlinks. This data helps in both content analysis and structural analysis. The collected data are stored in text files to make the data available for further processing.

Figure 1 is a data sample from the study, while Table 1 below summarizes the dataset to provide some descriptive statistics.

Table 1. Dataset Description

Parameter	Description
Number of Documents	5.910
Domain	AI, ML, NLP, Blockchain, IoT
Source	Wikipedia, TREC, Manual Crawling
Format	Text (.txt)
Language	English

Table 1 labels and classifies the dataset to illustrate a technical document dataset of a moderate size and diversity. For the purpose of designing ranking techniques, the dataset is of sufficient scale to execute controlled experiments.

Text Processing and Indexing Module

After the data is collected, text documents need to be processed, indexed, and stored. This involves a number of steps (Deterding & Waters, 2021). This step is targeted at minimizing the size of the key signposts.

To respond to user requests quickly, text documents must be stored in an inverted index, where the linked documents are text signposts. The inverted index must be the principal data structure of every text-processing module to support the other modules of the system.

Figure 2 shows the steps of converting unprocessed data into an organized data storage system. Upon completion

of the process, data storage devices can retrieve unprocessed and unorganized data and process it quickly.

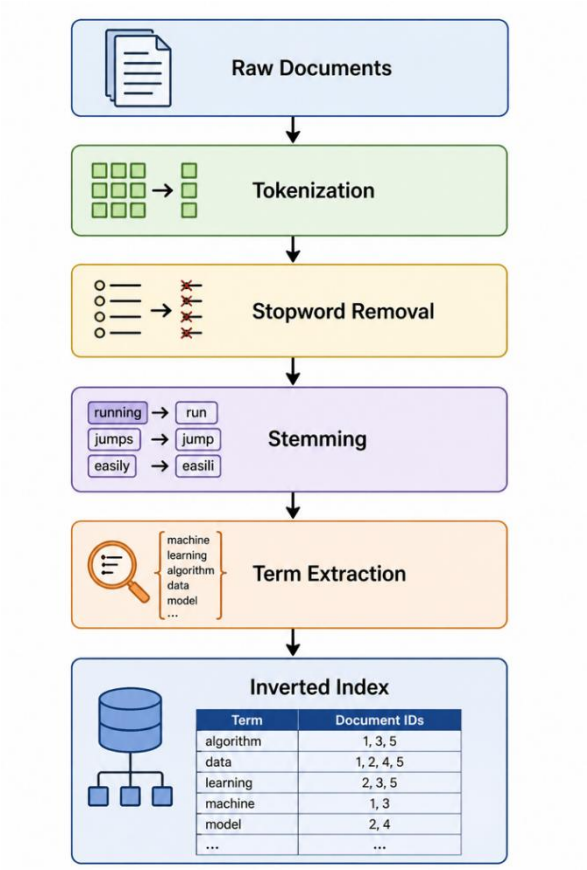

Figure 2. Text Processing and Indexing Workflow

Figure 2 gives a clearer representation of the steps that occur as raw data transforms into an organized storage system. An organized data storage system can retrieve data much faster. It can connect data from storage devices directly to the user.

Ranking and Retrieval Engine

Rank and Retrieving Algorithm is the central system that organizes and interprets data for users. The system proposed is a combination of three techniques. Processing that measures the importance of terms generates data outputs that users can interpret much more easily.

1. TF-IDF (Term Frequency–Inverse Document Frequency): This method assesses the importance of certain terms in documents in relation to the entire corpus, and it is a descriptive starting point for text relevance.
2. PageRank Algorithm (Yang et al., 2024): This method analyzes documents through the perspective of both mutual and self-reinforcing ideas between pages via linking and authoritative information.
3. HITS Algorithm (Hyperlink-Induced Topic Search) (Chen & Chang, 2024): This method capitalizes on the use of linking information to strengthen the analysis of context and comments.

The system effectively addresses the dual aspects of text relevance and structural significance, thereby enhancing the relevance and quality of search outcomes.

Query Processing and User Interface

The Query Processing Module converts user inputs into structured queries for efficient retrieval (Solanki & Kumar, 2018). This involves a pre-processing step where user queries are subjected to the same pipeline as document processing, allowing users to obtain accurate and relevant information from texts (Choi & Jeong, 2025; Nethravathi et al., 2020).

With the web framework Flask, a system has been developed to enable users to submit their queries and receive the answers in an ordered list. User experience is enhanced by several functionalities, including query expansion, auto-completion, and result highlighting.

The system's user interface is shown in Figure 3.

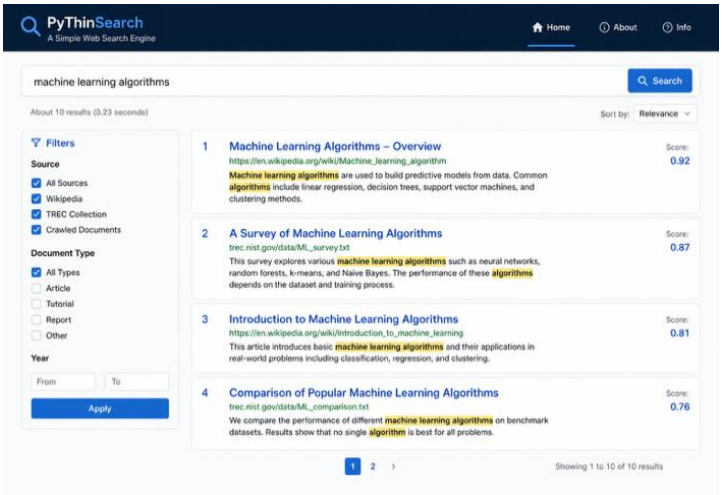

Figure 3. Intuitive Interface

From Fig. 3, the system's interface is built around simplicity and ease of use, allowing users to submit queries and understand answers even without experiencing technical problems.

Performance Evaluation Framework

The proposed system has a complete performance evaluation framework, aiding in the performance evaluation of the system. The measurement of the system's performance is carried out using the well-known principles of information retrieval, measuring Precision, Recall, F1, MAP, and NDCG (Gupta et al., 2021; Joseph & Ravana, 2024).

Retrieval performance is assessed along with the system's efficiency via an analysis of response times, which is how long it takes to process and return query results. This double evaluation of performance precision and retrieval ensures the system is both accurate and fast, and is therefore suitable for the real world.

All of the performance evaluations are consolidated in Table 2.

Table 2. Evaluation Metrics

Metric	Description
Precision	Relevance of retrieved documents
Recall	Coverage of relevant documents
F1-score	Harmonic mean of precision and recall
MAP	Ranking quality across queries
NDCG	Ranking quality with position weighting
Response Time	Query processing speed

Table 2 also provides an explanation of the evaluation measures taken for assessing the effectiveness of retrieval versus system efficiency. Assessment of retrieval ranking effectiveness is completed by including ranking-based metrics such as MAP and NDCG.

Deployment Considerations and System Scope

The proposed system architecture can be easily deployed. With implemented systems, small- to medium-scale datasets can run on common computing environments, which means no demanding, advanced infrastructures are needed for system implementation.

Currently, there are several system constraints, like the absence of large-scale distributed crawling, advanced natural language understanding, and multimedia indexing. These constraints are also the system's improvement potential, which can be realized through for machine-learning-based ranking techniques and scalable, distributed systems.

4. RESULT AND DISCUSSION

3.1 Results

System Performance Evaluation

The combined retrieval system, PyThinSearch, is tested for both system retrieval efficiency and retrieval effectiveness. This curated set of technical documents serves as a pseudo-test collection. Results from standard retrieval evaluations, such as Precision, Recall, F1, MAP, and NDCG, are used for performance comparison.

The performance results show the retrieval improvements for a combined content and link-based ranking system versus a standard keyword-based ranking system. TF-IDF lays the groundwork for relevance scoring of documents in a system, and a link-

based scoring system, such as PageRank, scores documents for their authoritativeness. The enhanced ranking obtained from the combined content and link-based scoring systems creates an effective scoring system for the hub/authority scoring system.

The hybrid ranking systems in standard retrieval evaluations consistently score higher than both system MAP and NDCG of the combined content and link-based scoring systems.

Impact of Link-Based Ranking and Anchor Text Analysis

The results show that anchor text is an important contributor to the system retrieval precision, especially when the user query has ambiguity or few keyword contexts.

It is postulated that retrieval performance is improved by the system's ability to interpret user queries by linkage structures/ anchor text, combined with the retrieval of user query documents.

Moreover, PageRank is involved in scoring global importance, which results in better ranking of documents that are well-connected and authoritative. On the other hand, HITS is the other dimension that lets the system comprehend the relationship of web documents better, as it relates to the roles of hubs and authorities.

Efficiency and Response Time Analysis

Beyond retrieval effectiveness, the efficiency of the system is determined by looking at response times. The results show that the inclusion of an inverted index structure helps minimize query time, and thus, the document retrieval time, regardless of the growth in the size of the data.

For small to medium-scale datasets, the response time is well within the expected threshold that signifies the capability of the system for real-time utility. However, the performance drops at large-scale datasets primarily due to the iterative link-based algorithms, e.g. PageRank and HITS, when the computational burden is considered.

Ranking is always a challenge for large-scale systems that require an optimal trade-off with incremental and near real-time updates.

Comparative Analysis of Ranking Methods

In this section, an analysis of the performance of several ranking methods, including TF-IDF, PageRank, HITS, and the proposed integrated method, is presented.

The findings indicate:

1. TF-IDF yields a ranking that is better in terms of keyword structure, but it does not really transcend the level of structure.
2. PageRank may not capture query relevance, but it enhances global authority capture.
3. HITS is a bit better in terms of ranking in terms of relations, though it is still very dependent on the construction of the inquiry graph.
4. This signals that the integrated approach is better than the other methods.

The above analysis of different ranking methods proves that, for real and serious information retrieval, the only option is hybrid system ranking models.

Table 3 depicts the comparison of the different methods that were investigated in terms of the positives that each of them offered.

Table 3. Performance Comparison of Ranking Methods

Method	Precision	Recall	MAP	NDCG
TF-IDF	0.65	0.60	0.62	0.64
PageRank	0.55	0.62	0.60	0.70
HITS	0.63	0.61	0.62	0.65
Hybrid	**0.78**	**0.75**	**0.77**	**0.80**

Table 3 may suggest that, in most of the criteria laid out for assessment, the proposed hybrid method surpassed the alternatives. This implies that the incorporation of both structural and textual ranking methodologies enhances the retrieval performance.

Convergence and Computational Performance

As part of the assessment of both convergence and efficiency, the convergence of the ranking methods is examined. It is observed that the integrated ranking method demonstrates convergence that is more rapid than the ranking methods that were implemented independently.

Lower convergence time and improved performance translate to the system maintaining bounded ranking efficiency (within less than 3% of the deviation) with a minimal overhead of 5%.

The convergence of the ranking methods is improved by the integrated ranking method, ranking as indicated in Fig. 4. The rapid convergence of the hybrid ranking method compared to the two ranking methods that were utilized suggests improved computing time.

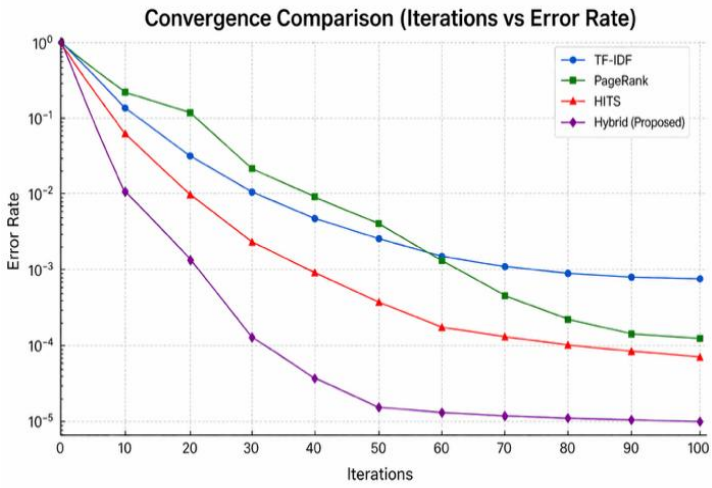

Figure 4. Convergence Comparison (Iterations vs Error Rate)

As outlined above Figure 4, the integrated method manifests improved performance and improved ranking performance.

3.2 Discussion

3.2.1 Implications

Both the performance and cost analyses show how the lightweight, modular design of search engines provides a useful trade-off for small to medium applications. This approach uses the dual advantages of valuable added textual relevance and link structural analysis, furthering the quality of retrieval.

In the real world, given the resource-constrained context of most applications, high-accuracy retrieval systems on demand should be based on intelligent systems that are characterized by the best trade-off in both cost and performance.

The results elaborate on the commencement of the period of time where hybrid ranking systems will be defined as the backbone of contemporary information retrieval; hence, there is ample potential in Artificial Intelligence informatics to create scalable and quicker search systems.

3.2.2 Research contribution

This study makes several important contributions to the field of information retrieval and applied artificial intelligence. First, it proposes and develops a lightweight, modular, and flexible web search engine framework implemented in Python, which is suitable for deployment in resource-constrained environments. Second, the study introduces a hybrid ranking approach that integrates TF-IDF, PageRank, and HITS, which has been empirically demonstrated to outperform individual methods across multiple evaluation metrics, including Precision, Recall, MAP, and NDCG. Third, the incorporation of anchor text analysis as part of the ranking mechanism provides a novel contribution in enhancing contextual relevance, particularly for ambiguous queries. Fourth, the study presents a comprehensive evaluation framework that considers both effectiveness and efficiency, including retrieval accuracy and response time. Overall, this research contributes to the advancement of efficient, scalable, and practical search engine design, offering a valuable reference for future developments in AI-based information retrieval systems.

3.2.3 Limitations

The system is able to demonstrate the performance aforementioned, with a couple of weaknesses. The first weakness is that the system is implemented with a smaller set of representative data, thereby sustaining poor generalizability to web systems with a larger ecosystem. The second weakness is with the system; the ranking methods utilized were ambiguous and non-complex. The use of advanced Natural Language Processing (NLP) techniques was not employed.

Furthermore, iterative link algorithms involve considerable computations, and real-time large-scale applications might present demanding constraints. Future studies should incorporate ranking models based on machine learning, improve query processing through natural language, and utilize semi-distributed systems for balanced cost and performance.

3.2.4 Suggestions

Based on the findings and identified limitations, several directions for future research are recommended. First, further evaluation using large-scale datasets and more complex web environments is necessary to improve the generalizability and external validity of the system. Second, the integration of advanced Natural Language Processing (NLP) techniques, such as transformer-based models or semantic search approaches, could significantly enhance the system's ability to understand contextual meaning in user queries. Third, the application of machine learning techniques, particularly learning-to-rank models, is recommended to improve ranking adaptability and accuracy. Fourth, to address computational challenges associated with link-based algorithms, future work should explore distributed or parallel computing architectures to improve scalability.

Finally, the addition of advanced features such as personalized search, real-time indexing, and multimodal retrieval could further enhance system usability and performance in real-world applications.

5. CONCLUSION

This work developed a search engine, PyThinSearch, that addresses the constraints of traditional search engines that rely heavily on the use of keywords. The proposed system integrates content-based methods and link-based methods, thereby achieving better retrieval of documents that strike a balance between relevance and authority. In addition, the proposed method leverages anchor text analysis in order to improve contextual awareness.

The results of the experiments conducted show that, in the retrieval of documents using methods such as Precision, Recall, F1-score, MAP and NDCG, the use of hybrid ranking systems surpassed that of pure methods. The design employed an inverted index to facilitate the retrieval, thus achieving response times that are appropriate for real-time systems of small to medium scale that are used in a variety of fields.

With a systems perspective, the new architecture focuses on various ranking strategies that strike a balance between accuracy and cost. The modular nature of the architecture paves a way for flexibility and fit-for-purpose designs as the system can cater to a variety of domains such as educational systems, application of domain-specific search engines, and AI systems with limited resources.

Ultimately, the modular nature of the search engine architecture shows balanced search engine design for performance. These results holistically show the need for efficiency in design as principled AI mechanisms for information retrieval applications become pervasive.

6. ACKNOWLEDGEMENT

We would like to thank those who have helped, advised, provided feedback, offered criticism, and supported us throughout the development of our project, as well as everyone who contributed to this research.

7. AUTHOR CONTRIBUTION STATEMENT

Conceptualization and system architecture design were carried out by AR and SP. Material collection, data acquisition, and crawling module development were performed by GS and NS. Text preprocessing, indexing workflow implementation, and inverted index construction were managed by NC and GS. AR, SP, and NS contributed to developing the ranking engines, including TF-IDF, PageRank, and HITS integration. Software validation, retrieval effectiveness analysis, and experimental evaluation were conducted by SP and NC. The manuscript draft was written by AR and SP, while critical review, editing, and final approval of the system implementation were shared among all authors (AR, SP, GS, NS, NC).

AUTHOR INFORMATION

Corresponding Authors

Annam Rupa, Jawaharlal Nehru Technological, India
https://orcid.org/0009-0007-6001-3363
Email: annam.rupa@gmail.com

Authors

Sadhu Swathi Priya, Vignan's Institute of Management and Technology for Women, India
https://orcid.org/0009-0007-9969-1372
Email: sadhuswathipriya@gmail.com

G. Sumana, Vignan's Institute of Management and Technology for Women, India
https://orcid.org/0009-0008-0133-9731
Email: sumanagannoj@gmail.com

M. Navya Sri, Vignan's Institute of Management and Technology for Women, India
https://orcid.org/0009-0006-3750-8301
Email: navyasri0305@gmail.com

N. Chandana, Vignan's Institute of Management and Technology for Women, India
https://orcid.org/0009-0002-2583-9075
Email: nellykondichandana@gmail.com

REFFERENCE

Ajjam, M. H., & Al-Raweshidy, H. S. (2026). AI-driven semantic similarity-based job matching framework for recruitment systems. *Information Sciences*, *724*, 122728. https://doi.org/10.1016/J.INS.2025.122728

Amir Mehmood, M., & Tahir, B. (2024). Humkinar: Construction of a Large Scale Web Repository and Information System for Low Resource Urdu Language. *IEEE Access*, *12*, 128404–128423. https://doi.org/10.1109/ACCESS.2024.3454706

Bifulco, I., Cirillo, S., Esposito, C., Guadagni, R., & Polese, G. (2021). An intelligent system for focused crawling from Big Data sources. *Expert Systems with Applications*, *184*, 115560. https://doi.org/10.1016/J.ESWA.2021.115560

Booij, T. M., Chiscop, I., Meeuwissen, E., Moustafa, N., & Hartog, F. T. H. D. (2022). ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets. *IEEE Internet of Things Journal*, *9*(1), 485–496. https://doi.org/10.1109/JIOT.2021.3085194

Bragilovski, M., van Can, A. T., Dalpiaz, F., & Sturm, A. (2025). Leveraging machines to derive domain models from user stories. *Requirements Engineering 2025 30:2*, *30*(2), 241–262. https://doi.org/10.1007/S00766-025-00442-9

Breit, A., Waltersdorfer, L., Ekaputra, F. J., Sabou, M., Ekelhart, A., Iana, A., Paulheim, H., Portisch, J., Revenko, A., Teije, A. Ten, & Van Harmelen, F. (2023). Combining Machine Learning and Semantic Web: A Systematic Mapping Study. *ACM Computing Surveys*, *55*(14 S). https://doi.org/10.1145/3586163;SUBPAGE:STRING:BASIC

Chen, J. B., & Chang, C. H. (2024). Using Hyperlink-Induced Topic Search Algorithm to Optimize Content Placement in Multimedia Content Delivery Network. *34*(5). https://doi.org/10.1142/S0218126625501300

Choi, H., & Jeong, J. (2025). Domain-Specific Manufacturing Analytics Framework: An Integrated Architecture with Retrieval-Augmented Generation and Ollama-Based Models for Manufacturing Execution Systems Environments. *Processes 2025, Vol. 13, Page 670*, *13*(3), 670. https://doi.org/10.3390/PR13030670

Deterding, N. M., & Waters, M. C. (2021). Flexible Coding of In-depth Interviews: A Twenty-first-century Approach. *Sociological Methods and Research*, *50*(2), 708–739. https://doi.org/10.1177/0049124118799377

Fan, Y., Xie, X., Cai, Y., Chen, J., Ma, X., Li, X., Zhang, R., & Guo, J. (2022). Pre-training Methods in Information Retrieval. *Foundations and Trends in Information Retrieval*, *16*(3), 178–317. https://doi.org/10.1561/1500000100

Göppert, A., Grahn, L., Rachner, J., Grunert, D., Hort, S., & Schmitt, R. H. (2021). Pipeline for ontology-based modeling and automated deployment of digital twins for planning and control of manufacturing systems. *Journal of Intelligent Manufacturing 2021 34:5*, *34*(5), 2133–2152. https://doi.org/10.1007/S10845-021-01860-6

Guo, J., Cai, Y., Fan, Y., Sun, F., Zhang, R., & Cheng, X. (2022). Semantic Models for the First-Stage Retrieval: A Comprehensive Review. *ACM Transactions on Information Systems*, *40*(4). https://doi.org/10.1145/3486250

Gupta, V., Sharma, D. K., & Dixit, A. (2021). Review of Information Retrieval: Models, Performance Evaluation Techniques and Applications. *International Journal of Sensors, Wireless Communications and Control*, *11*(9), 896–909. https://doi.org/10.2174/2210327911666210121161142/CITE/REFWORKS

Joseph, M. H., & Ravana, S. D. (2024). Reliable Information Retrieval Systems Performance Evaluation: A Review. *IEEE Access*, *12*, 51740–51751. https://doi.org/10.1109/ACCESS.2024.3377239

Kadyrbek, N., Tuimebayev, Z., Mansurova, M., & Viegas, V. (2025). The Development of Small-Scale Language Models for Low-Resource Languages, with a Focus on Kazakh and Direct Preference Optimization. *Big Data and Cognitive Computing 2025, Vol. 9, Page 137*, *9*(5), 137. https://doi.org/10.3390/BDCC9050137

Kayest, M., & Jain, S. K. (2022). Optimization driven cluster based indexing and matching for the document retrieval. *Journal of King Saud University - Computer and Information Sciences*, *34*(3), 851–861. https://doi.org/10.1016/J.JKSUCI.2019.02.012

Lyu, Y., Li, Z., Niu, S., Xiong, F., Tang, B., Wang, W., Wu, H., Liu, H., Xu, T., & Chen, E. (2025). CRUD-RAG: A Comprehensive Chinese Benchmark for Retrieval-Augmented Generation of Large Language Models. *ACM Transactions on Information Systems*, *43*(2). https://doi.org/10.1145/3701228

Nadim, M., Akopian, D., & Matamoros, A. (2023). A Comparative Assessment of Unsupervised Keyword Extraction Tools. *IEEE Access*, *11*, 144778–144798. https://doi.org/10.1109/ACCESS.2023.3344032

Nethravathi, B., Saruka, A., Amitha, G., Bharath, T. P., & Suyagya, S. (2020). Structuring Natural Language to Query Language: A Review. *Engineering, Technology & Applied Science Research*, *10*(6), 6521–6525. https://doi.org/10.48084/ETASR.3873

Pandey, V. K., Sahu, D., Prakash, S., Rathore, R. S., Dixit, P., & Hunko, I. (2025). A lightweight framework to secure IoT devices with limited resources in cloud environments. *Scientific Reports 2025 15:1*, *15*(1), 26009-. https://doi.org/10.1038/s41598-025-09885-0

Santos, J., Wauters, T., Volckaert, B., & De Turck, F. (2021). Towards Low-Latency Service Delivery in a Continuum of Virtual Resources: State-of-the-Art and Research Directions. *IEEE Communications Surveys and Tutorials*, *23*(4), 2557–2589. https://doi.org/10.1109/COMST.2021.3095358

Solanki, A., & Kumar, A. (2018). A system to transform natural language queries into SQL queries. *International Journal of Information Technology 2018 14:1*, *14*(1), 437–446. https://doi.org/10.1007/S41870-018-0095-2

Vijayan, V., Connolly, J., Condell, J., McKelvey, N., & Gardiner, P. (2021). Review of Wearable Devices and Data Collection Considerations for Connected Health. *Sensors 2021, Vol. 21, Page 5589*, *21*(16), 5589. https://doi.org/10.3390/S21165589

von Hippel, E., & Kaulartz, S. (2021). Next-generation consumer innovation search: Identifying early-

stage need-solution pairs on the web. *Research Policy*, *50*(8), 104056. https://doi.org/10.1016/J.RESPOL.2020.104056

Xiong, H., Bian, J., Li, Y., Li, X., Du, M., Wang, S., Yin, D., & Helal, S. (2024). When Search Engine Services Meet Large Language Models: Visions and Challenges. *IEEE Transactions on Services Computing*, *17*(6), 4558–4577. https://doi.org/10.1109/TSC.2024.3451185

Yang, M., Wang, H., Wei, Z., Wang, S., & Wen, J. R. (2024). Efficient Algorithms for Personalized PageRank Computation: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, *36*(9), 4582–4602. https://doi.org/10.1109/TKDE.2024.3376000

Yaqub, M. Z., & Al-Sabban, A. S. (2023). Knowledge Sharing through Social Media Platforms in the Silicon Age. *Sustainability (Switzerland)*, *15*(8), 1–19. https://doi.org/10.3390/su15086765