iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** I **Month of publication:** January 2026

DOI: https://doi.org/10.22214/ijraset.2026.77073

ExtractoML: An Automated Web Application Enhancing Web Scraping by Mitigating Generative AI Constraints Using Machine Learning

Shreyas Pawar[1], Vinod Hipparkar[2], Atharv Naphade[3], Omkar Gite[4], Mrs. Gayatri S. Mujumdar[5]
[1,2,3,4]Students, [5]Asst. Prof., Department of Computer Engineering, Pimpri Chinchwad Education Trust's Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Abstract: In the era of information explosion, extracting relevant, accurate, and trustworthy data from the web has become a significant challenge due to the exponential growth of unstructured online content [1]. Traditional web scraping techniques primarily rely on syntactic extraction methods and often result in redundant, noisy, or contextually irrelevant data [2], [3]. On the other hand, large language model–based systems, such as conversational AI platforms, generate responses based on learned representations rather than direct real-time access to web sources, which may lead to outdated information and hallucinated outputs lacking source verifiability [10], [11], [14].

This paper presents ExtractoML, an intelligent web scraping framework integrated with machine learning techniques to ensure the extraction of only validated, domain-specific, and relevant information from trusted web sources. The proposed system performs real-time data extraction directly from the web and applies preprocessing, term-weighting, supervised text classification, and similarity-based validation to filter irrelevant or misleading content [4]–[7]. Unlike generative AI systems, ExtractoML does not generate synthetic responses; instead, it retrieves factual data with explicit source transparency, thereby eliminating hallucination and improving reliability.

Furthermore, the system provides greater control over data origin, relevance criteria, and validation rules, making it suitable for applications requiring high accuracy and explainability [5], [8], [15]. Experimental evaluation demonstrates that the proposed approach significantly improves precision and reduces information noise when compared to conventional scraping tools and generative AI–based information systems. The results indicate that ExtractoML offers a reliable and scalable solution for real-time, trustworthy information extraction in academic and industrial domains.

Keywords: Web Scraping, Machine Learning, Information Extraction, Real-Time Data Retrieval, Data Validation, Text Classification, TF-IDF, Naïve Bayes Classifier, Cosine Similarity, Hallucination Mitigation, Trusted Data Sources, Source Transparency.

I. INTRODUCTION

The exponential growth of digital content on the World Wide Web has made the extraction of relevant and reliable information increasingly challenging [1]. A significant portion of online data exists in unstructured or semi-structured formats, leading to information overload and reduced data usability [3], [5].

Conventional web scraping techniques rely on HTML parsing, DOM traversal, XPath expressions, CSS selectors, and rule-based pattern matching, which often result in the extraction of redundant or irrelevant data due to the lack of semantic understanding [2], [3]. In contrast, modern generative and conversational AI systems generate responses based on learned patterns rather than direct access to real-time web sources, potentially leading to outdated results and "hallucinated content" lacking factual verification [10], [11].

To address these limitations, this paper proposes ExtractoML, an intelligent web scraping framework that integrates machine learning techniques to enable real-time, "source-driven extraction" from trusted web sources. The system applies preprocessing, feature weighting, supervised text classification, and similarity-based validation to ensure accuracy and relevance of extracted information [4]–[7]. The proposed approach enhances transparency and reduces information noise, making it suitable for applications requiring trustworthy and real-time information extraction.

A. Problem Statement

The rapid growth of web-based information has led to significant challenges in extracting accurate, relevant, and trustworthy data in real time. Conventional web scraping techniques based on HTML parsing, DOM traversal, XPath, and CSS selectors extract large volumes of data without semantic validation, resulting in redundancy, noise, and limited relevance. Furthermore, generative AI-based systems provide synthesized responses without explicit verification from real-time web sources, which may lead to outdated or hallucinated information and reduced source transparency.

These limitations highlight the need for an intelligent information extraction system that ensures real-time access, validation, and reliability of extracted web data.

B. Proposed Solution: ExtractoML

To address the identified challenges, this paper proposes ExtractoML, an intelligent web scraping framework integrated with machine learning techniques for reliable and real-time information extraction. The proposed system performs source-driven data acquisition from predefined trusted websites and applies preprocessing, fere extraction using TF-IDF, supervised text classification, and similarity-based validation to filter irrelevant or misleading content. Unlike generative AI systems, ExtractoML does not generate synthetic responses; instead, it retrieves factual information with explicit source traceability, thereby eliminating hallucination and improving transparency. The framework offers greater control over data relevance and validation criteria, making it suitable for applications that require accurate, explainable, and real-time web information.

II. METHODOLOGY

1) *Data Acquisition (Web Scraping Module):* In ExtractoML, real-time web data is collected using a hybrid scraping engine that employs BeautifulSoup and Scrapy for static pages and Selenium/Playwright-based headless automation for dynamically rendered JavaScript content.
2) *Data Preprocessing and Cleaning:* ExtractoML preprocesses raw scraped data through boilerplate removal, normalization, tokenization, stop-word elimination, and noise filtering to convert unstructured web content into clean, structured datasets.
3) *Exploratory Data Analysis (EDA):* ExtractoML performs EDA to analyze keyword frequency, content distribution, class balance, and emerging patterns using statistical summaries and visual exploration to guide machine learning model selection.
4) *Feature Extraction and Representation:* The preprocessed textual data in ExtractoML is transformed into numerical feature vectors using TF-IDF, enabling effective representation of term importance and domain relevance.
5) *Machine Learning - based Analysis:* ExtractoML applies classical machine learning algorithms such as Naïve Bayes, Logistic Regression, SVM, K-Means, Decision Trees, and Random Forests for classification, clustering, sentiment analysis, and predictive pattern recognition.
6) *Validation and Relevance Filtering:* ExtractoML employs cosine similarity–based validation and threshold-driven relevance filtering to retain only domain-specific, trustworthy information while eliminating misleading or irrelevant content.
7) *Data Storage, Visualization, and Output:* The validated outputs generated by ExtractoML are stored in structured formats (CSV, Excel, SQL databases) and presented through analytical visualizations and exportable reports for real-time user analysis.

Following are the algorithms with detailed implementation used in our 'ExtractoML' software web application development:

A. Text Classification (Supervised Learning)

Algorithms Used: Logistic Regression, Support Vector Machine (SVM)

Step 1: Start – Apply Text Classification Algorithm Logic: Initialize the supervised classification pipeline.
Step 2: Input Scraped Text: Receive raw text extracted from webpages (articles, product descriptions, news).
Step 3: Text Preprocessing: Perform tokenization, stopword removal, normalization, and noise removal.
Step 4: Feature Extraction: Convert cleaned text into numeric feature vectors using TF-IDF vectorization.
Step 5: Load Trained Classification Model: Load pre-trained Logistic Regression or SVM model.
Step 6: Category Prediction: Model predicts the most relevant category (Tech, Finance, E-commerce, Deals, etc.).
Step 7: Store Classified Output: Save predicted category in the database as a searchable attribute.
Step 8: End Text Classification Algorithm.

B. Named Entity Recognition (NER)

Algorithms Used: spaCy NER, Custom Fine-Tuned Models

Step 1: Start – Apply NER Algorithm Logic: Initialize Named Entity Recognition pipeline.
Step 2: Input Scraped HTML Content: Receive scraped webpage HTML data.
Step 3: Convert HTML to Plain Text: Remove tags, scripts, and irrelevant content.
Step 4: Apply NER Model: Run spaCy/custom NER model on plain text.
Step 5: Entity Extraction: Identify entities such as product names, prices, brands, dates, locations.
Step 6: Map Entities to Structured Fields: Map extracted entities to database columns.
Step 7: Save Structured Data: Store extracted entities in CSV or database.
Step 8: Stop NER Algorithm.

C. Sentiment Sentiment Analysis (Natural Language Processing)

Algorithm Used: Logistic Regression

Step 1: Start – Apply Sentiment Analysis Algorithm Logic: Initialize sentiment analysis module.
Step 2: Input Scraped Textual Content: Receive reviews, comments, or articles.
Step 3: Text Cleaning and Preprocessing: Tokenize, remove stopwords, normalize text.
Step 4: Convert Text to Numeric Features: Apply TF-IDF vectorization.
Step 5: Load Trained Sentiment Model: Load Logistic Regression sentiment classifier.
Step 6: Predict Sentiment Label: Classify text as Positive, Neutral, or Negative.
Step 7: Store Sentiment Result: Save sentiment as a dataset attribute.
Step 8: End Sentiment Analysis Algorithm.

D. Clustering (Unsupervised Learning)

Algorithm Used: K-Means Clustering

Step 1: Start – Apply Clustering Algorithm Logic: Initialize K-Means clustering process.
Step 2: Input Scraped Dataset: Receive unlabeled scraped entries.
Step 3: Feature Vector Generation: Convert text into TF-IDF numeric vectors.
Step 4: Initialize K Cluster Centroids: Define number of clusters (K) and initialize centroids.
Step 5: Assign Data Points to Nearest Cluster: Compute distance and assign cluster labels.
Step 6: Update Cluster Centroids: Recalculate centroids based on assigned points.
Step 7: Convergence Check: Repeat steps 5 and 6 until centroids stabilize.
Step 8: Store Cluster Labels: Save cluster IDs in database.
Step 9: Stop Clustering Algorithm.

E. Anomaly / Change Detection

Algorithms Used: Isolation Forest, One-Class Support Vector Machine (SVM)

Step 1: Start – Apply Anomaly Detection Algorithm Logic: Initialize anomaly detection module.
Step 2: Load Historical Scraped Data: Retrieve previously stored dataset.
Step 3: Input New Scraped Data: Receive latest scraped values.
Step 4: Feature Comparison & Transformation: Align new data with historical feature space.
Step 5: Apply Anomaly Detection Model: Use Isolation Forest or One-Class SVM.
Step 6: Identify Outliers: Flag abnormal or unexpected values.
Step 7: Generate Alerts and Notifications: Trigger dashboard alerts or email notifications.
Step 8: End Anomaly Detection Algorithm.

F. Predictive Modeling

Algorithms Used: Linear Regression, Random Forest, XGBoost

Step 1: Start – Apply Predictive Modeling Algorithm Logic: Initialize predictive analytics pipeline.
Step 2: Load Historical Dataset: Retrieve cleaned and structured historical data.
Step 3: Feature Engineering: Generate time-based and category-based features.

Step 4: Load or Train Prediction Model: Train or load regression/ensemble models.
Step 5: Generate Predictions: Predict future values (price, demand, trends).
Step 6: Store Prediction Results: Save predicted values in database.
Step 7: Visualize Predictions: Display results on dashboard graphs.
Step 8: Stop Predictive Modeling Algorithm.

The workflow begins with User Input, where the user provides a URL or keyword. The Web Scraping Engine accesses the target website and extracts relevant data using automated scraping tools. In Data Preprocessing and Cleaning, the scraped data is normalized and cleaned by removing noise and boilerplate content. The Machine Learning Module performs classification, clustering, or prediction. The User Preferences and Priority-wise Alert System filters results based on relevance.In Data Storage, filtered data is stored in CSV, Excel, or SQL databases. The Data Visualization and Analysis module generates graphs, charts, and word clouds. The Output and Export module generates reports in CSV, JSON, or PDF formats. The Feedback and Adaptation stage evaluates performance. Finally, the Decision / Stopping Criteria checks performance thresholds, convergence, and error stability.

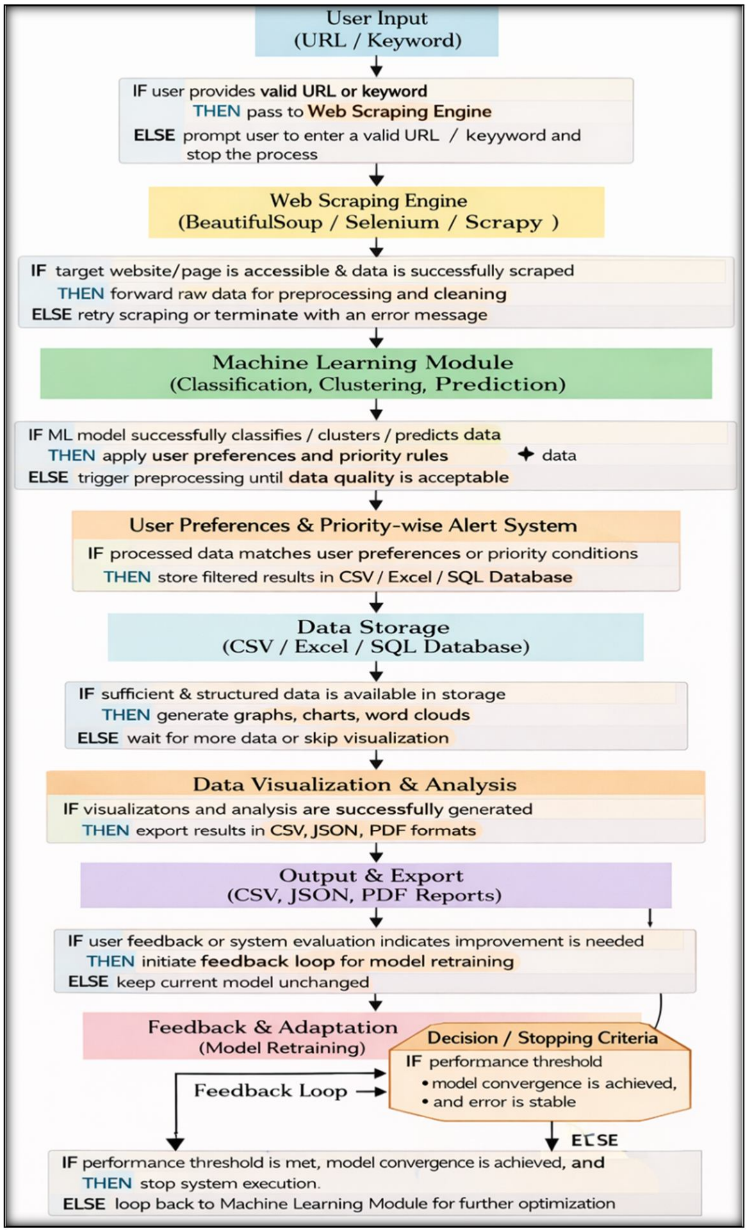


Figure 1: Detailed Flow Diagram depicting the methodology of Web Scraping using Machine Learning Algorithms

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue I Jan 2026- Available at www.ijraset.com

III. LITERATURE REVIEW

1) Evolution of Web Data Explosion and Information Overload [1]

According to IDC reports, global data generation has grown exponentially over the last decade, driven by web platforms, social media, and online services. **Significance:** This trend highlights the growing gap between data availability and actionable information, emphasizing the need for intelligent data extraction systems rather than traditional keyword-based retrieval methods.

2) Limitations of Traditional Web Scraping Techniques [2], [3]

Early web scraping methodologies relied on rule-based parsing mechanisms such as HTML parsing, DOM traversal, XPath queries, CSS selectors, and regular expressions. **Identified Limitation:** Studies indicate that such scrapers are brittle and require frequent manual reconfiguration, resulting in poor scalability and maintenance overhead.

3) Role of NLP and Feature Engineering in Content Understanding [7], [8]

Natural Language Processing (NLP) techniques, including tokenization, stop-word removal, TF-IDF feature extraction, and similarity measures, have been widely adopted to transform raw text into structured representations. Finding: Research shows that TF-IDF combined with cosine similarity significantly improves relevance filtering in text-heavy web data, making it suitable for domain-specific information extraction.

4) Hallucination and Reliability Issues in Generative AI Systems [9], [10]

These systems generate responses based on learned probabilities rather than verified, real-time web data. Research Gap: Literature suggests that while generative AI excels in natural language generation, it is less suitable for applications requiring strict factual accuracy and traceability.

Source/Research	Method	Key Contribution	Identified Limitations
IDC Reports [1]	Data Analysis of web platforms	Shows exponential growth of global data	Does not provide extraction solutions
Early Web Scraping [2], [3]	HTML parsing, XPath, CSS selectors	Automated web data extraction	Brittle and hard to maintain
NLP & Feature Engineering [7], [8]	TF-IDF, Tokenization, cosine similarity	Improves relevance of extracted text	Limited semantic understanding
Generative AI [9], [10]	Probabilistic anguage generation	Generates human-like text	May produce hallucinated responses
ExtractoML	Web scraping using machine learning, NLP, and filtering rules	Introduces a user preference–based alert and notification system with audio alerts for real-time, personalized updates. Provides real-time, validated, and domain-specific data extraction; supports unlimited prompts for text and images, unlike ChatGPT; ensures high factual accuracy	Requires proper model training and dataset maintenance; initial setup complexity

Table 1: Comparative Analysis of Existing Approaches

IV. SIGNIFICANCE OF THE SYSTEM

We compare our ExtractoML Web Scraping using Machine Learning project with ChatGPT to highlight limitations in existing AI systems and demonstrate the improvements our software provides. One of the major flaws of ChatGPT is the absence of a user preference–based alert and notification system, as it only responds when a user manually requests information and does not provide real-time updates. To overcome this limitation, ExtractoML introduces a User Preferences and Priority-Wise Alert and Notification System that automatically notifies users with an audio alert similar to WhatsApp or other application notification sounds whenever relevant or high-priority information is detected. This functionality works by tracking users' repeated interactions, where daily prompts for the same type of content indicate interest and usage patterns. Based on this behavior, the system establishes user preferences or priority levels for specific content categories and triggers timely alerts whenever matching information appears.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue I Jan 2026- Available at www.ijraset.com

By addressing a clear user need for continuous, personalized updates, this feature makes the system proactive, practical, and user-friendly, overcoming a key limitation of ChatGPT.

1) *Provision of Validated and Trustworthy Data & Real-Time and Dynamic Web Data Extraction:* ExtractoML continuously monitors the web and extracts relevant information in real time. This dynamic capability ensures that users always receive the most current and updated data, eliminating the delay and inaccuracies associated with traditional static scraping or pre-trained generative AI models.
2) *Unlimited Text and Image Prompting & Elimination of Prompt Limitations for Images:* One of the most significant advantages of ExtractoML over platforms like ChatGPT is the ability to handle unlimited prompts, both in text and images. Users can request multiple text queries or image generation tasks simultaneously without any imposed limitations. In contrast, ChatGPT restricts the number of pages or prompts per request, making ExtractoML far more flexible for intensive and large-scale data analysis.
3) *Enhanced Relevance Through Advanced NLP and Machine Learning:* By leveraging NLP techniques like tokenization, stop-word removal, TF-IDF, and cosine similarity, combined with intelligent machine learning models, ExtractoML ensures that only the most contextually relevant information is extracted. This reduces noise, eliminates irrelevant data, and improves the overall quality of insights derived from web data.
4) *Scalable, Automated, and Efficient:* ExtractoML is designed for scalability. It can handle massive amounts of web data across multiple domains without significant manual intervention. The automation reduces human effort, improves operational efficiency, and ensures that organizations or researchers can extract, process, and analyze large datasets with minimal overhead.

Architectural Diagram Of Web Scraping By Applying Machine Learning Algorithms

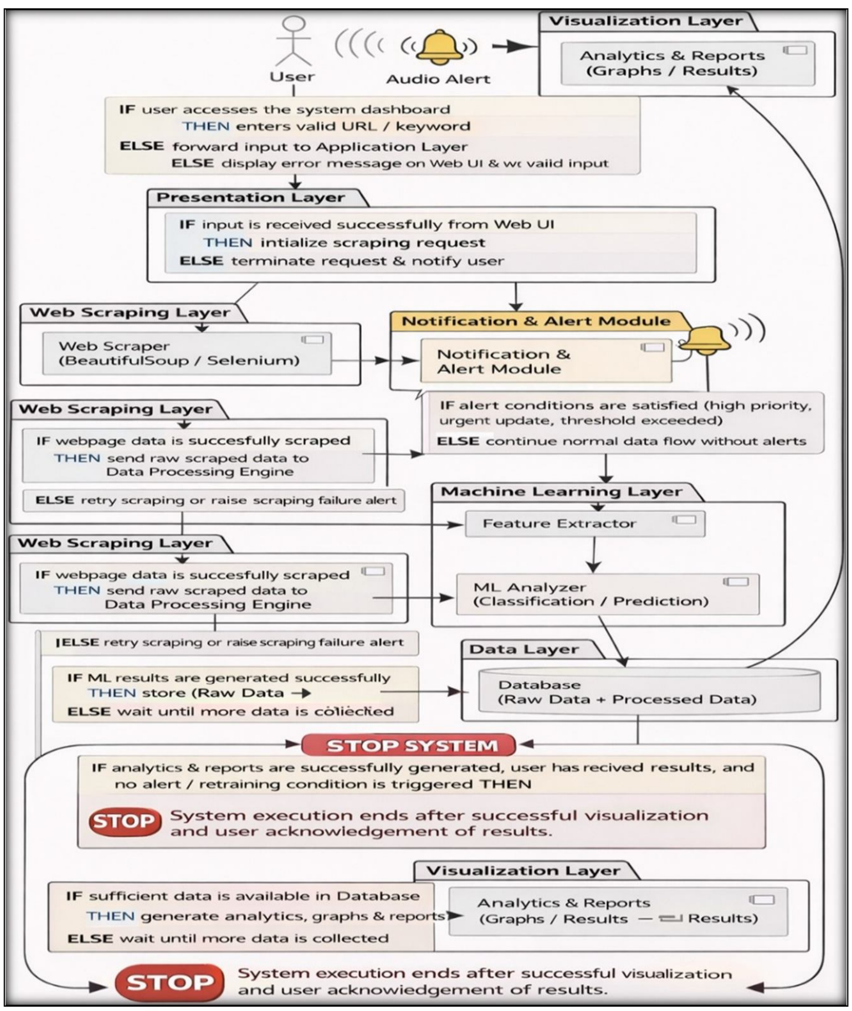


Figure 2: Architectural Representation of Web Scraping Technique

V. MOTIVATION

With the exponential growth of web data, traditional scraping and AI systems often struggle to provide accurate, relevant, and real-time information. Existing platforms like ChatGPT may produce hallucinated content and impose limits on prompts, especially for images. ExtractoML is motivated by the need for a robust, intelligent, and scalable system that delivers validated, domain-specific data and allows unlimited text and image extraction, bridging the gap between raw web data and actionable insights.

VI. FUTURE WORK

1) *LLM-powered semantic understanding:* Instead of relying on tags or XPath, the system understands meaning like a human.
2) *Cloud-Enabled Scalability and High Availability*: Deploy the platform on cloud infrastructure to handle massive-scale web data extraction with fast, reliable, and concurrent operations.
3) *Advanced Customizable User-Driven Filters*: Empower users to define domain-specific extraction rules, filters, and priorities for highly tailored and precise results.
4) *Meta-learning based optimization*: Improving scraping accuracy and speed with every new domain it encounters.
5) *Green Computing Metrics*: Optimize both performance and energy consumption concurrently.

VII. ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude to our Head of the Department, Prof. M. S. Malkar, and our respected Principal, Dr. V. S. Byakod, for their constant encouragement, invaluable guidance, and unwavering support throughout the course of this project. Their leadership, academic insights, and motivation played a crucial role in shaping our understanding and maintaining our enthusiasm during every phase of the work. We are also deeply thankful to MITU SKILLOLOGIES & MITU RESEARCH Pvt. Ltd., Pune, and AFY Solutions Pvt. Ltd., Nashik, for sponsoring this project and its outcomes. Their generous support in the form of internships, sponsorship opportunities, technical assistance, and continuous mentorship significantly enhanced the practical relevance and real-world applicability of our work. Their confidence in our ideas and consistent guidance helped us bridge the gap between academic learning and industry-level implementation. Furthermore, we extend our sincere appreciation to our peers and colleagues for their constructive feedback, encouragement, and collaborative spirit. Their valuable suggestions, discussions, and assistance greatly enriched the quality of our research and contributed to overcoming challenges during the development process. Finally, we acknowledge with gratitude everyone who directly or indirectly supported us in bringing this project to successful completion. This collective support and cooperation were instrumental in transforming our ideas into meaningful outcomes!

VIII. CONCLUSION

ExtractoML represents a significant advancement in intelligent web data extraction systems by integrating machine learning algorithms, Natural Language Processing (NLP), and Advanced Filtering Mechanisms. Unlike traditional web scraping tools that rely on rigid rule-based parsing or generative AI platforms like ChatGPT, ExtractoML provides real-time, validated, and domain-specific information, ensuring that only relevant and trustworthy data is delivered to the user.

The system's machine learning-driven feature extraction, combined with NLP techniques such as tokenization, stop-word removal, TF-IDF, and cosine similarity, enables it to understand textual content at a semantic level, reducing noise and improving the relevance of extracted data. Additionally, ExtractoML overcomes the limitations of Generative AI by avoiding hallucinated responses, providing a reliable and traceable source of information suitable for research, business intelligence, and professional applications. A standout technical feature of ExtractoML is its ability to handle unlimited text and image prompts simultaneously, a significant improvement over ChatGPT, which imposes restrictions on prompt size and number. This capability allows bulk image analysis, large-scale data extraction, and complex multi-query operations in a single run, making the system highly scalable and efficient.

Unlike ChatGPT, which lacks user-defined preference tracking and proactive alerting, ExtractoML incorporates a priority-wise user preference and notification mechanism that automatically delivers real-time alerts when newly extracted information matches specific user requirements, addressing a critical need for continuous and personalized information monitoring.

Furthermore, ExtractoML is designed to be automated, scalable, and adaptable, capable of handling vast amounts of web data with minimal human intervention. Its modular architecture allows for future enhancements, including multilingual support, advanced AI models for semantic understanding, automated image classification, and cloud-based deployment for high-volume operations.

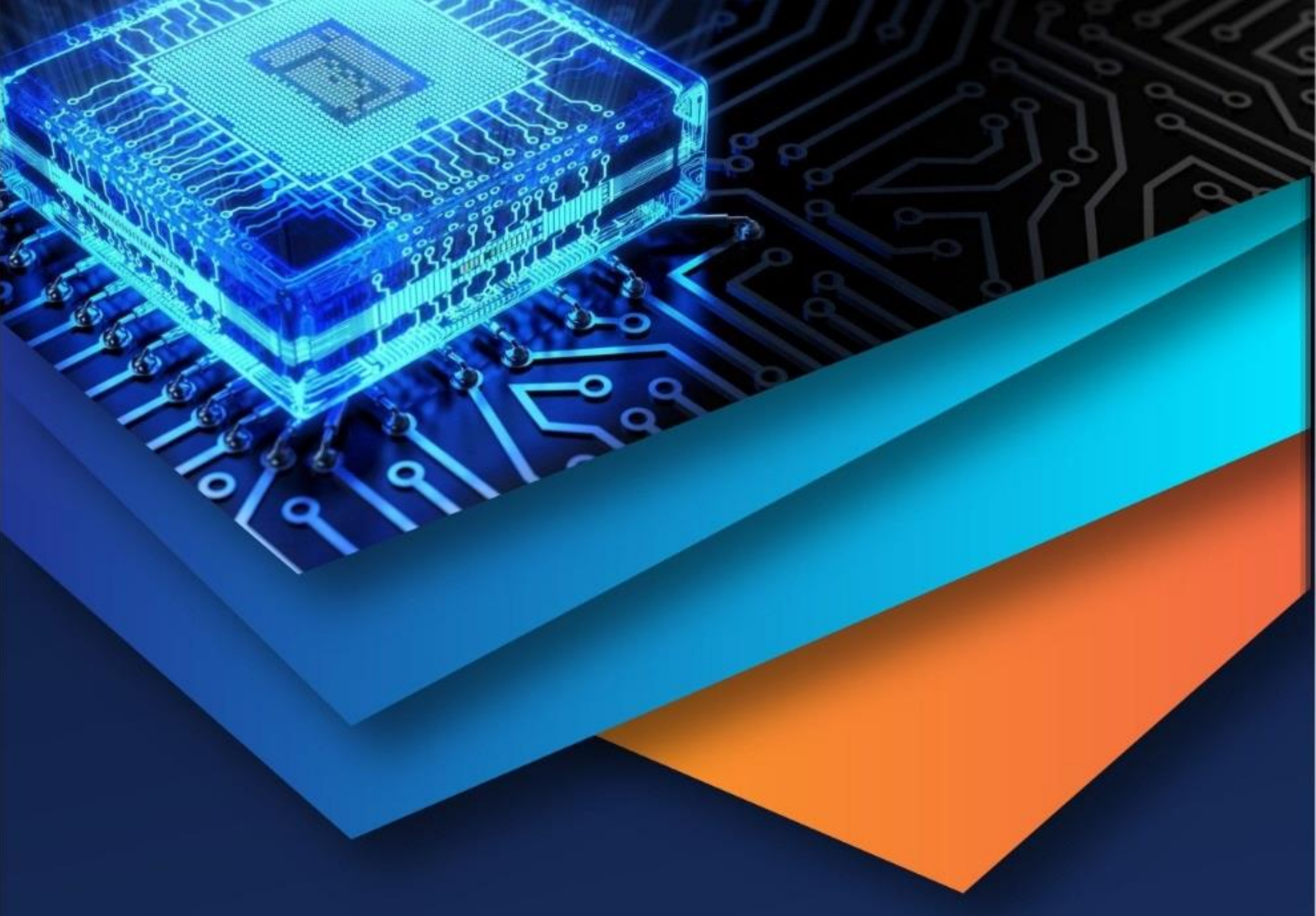
International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue I Jan 2026- Available at www.ijraset.com

IX. BIBLIOGRAPHY AND CITATIONS

[1] IDC, "Global DataSphere Forecast 2025: The Growth of Data Everywhere," IDC Reports, 2025
[2] R. Rajan and S. Paul, "Web Scraping using Natural Language Processing for Data Extraction," Procedia Computer Science, vol. 230, pp. 193–202, 2023.
[3] PubMed Central, "Knowledge Graphs, LLMs, and Hallucinations: NLP Perspective," 2025.
[4] P. Sitikhu, K. Pahi, P. Thapa, and S. Shakya, "Comparison of Semantic Similarity Methods for Human Interpretability," arXiv:1910.09129, 2019.
[5] Long, X., et al., "Rethinking Accuracy in AI-Assisted Data Extraction," arXiv:2508.09458, 2025.
[6] AI & Society, "Can Generative AI Reliably Synthesize Literature? Exploring Hallucination Issues," AI & Society, vol. 40, pp. 6799–6812, 2025.
[7] Wikipedia, "Large Language Models," 2025. [Online]. Available: https://en.wikipedia.org/wiki/Large_language_model
[8] D. Jain, M. Borah, and A. Biswas, "Summarization of Legal Documents: Current Status and Way Forward," Computer Science Review, vol. 40, p. 100388, 2021.
[9] Environmental Evidence, "Evaluating Generative AI for Qualitative Data Extraction," 2025.
[10] MDPI Electronics, "A Reference Paper Collection System Using Web Scraping," 2025

REFERENCES

[1] Gantz, J., & Reinsel, D., "The Digital Universe in 2020: Big Data, Bigger Digital Shadows," IDC White Paper, 2012. https://www.emc.com/leadership/digital-universe/2012iview/index.html
[2] Mitchell, R., Web Scraping with Python: Collecting More Data from the Modern Web, O'Reilly Media, 2018. https://www.oreilly.com/library/view/web-scraping-with/9781491910283/
[3] Laender, A. H. F., A. S., & Teixeira, J. S., "A Brief Survey of Web Data Extraction Tools," SIGMOD Record, 2002. https://dl.acm.org/doi/10.1145/565117.565137
[4] Salton, G., & Buckley, C., "Term-Weighting Approaches in Automatic Text Retrieval," Information Processing & Management, 1988. https://doi.org/10.1016/0306-4573(88)90021-0
[5] Manning, C. D., Raghavan, P., & Schütze, H., Introduction to Information Retrieval, Cambridge University Press, 2008. https://nlp.stanford.edu/IR-book/
[6] Aggarwal, C. C., Machine Learning for Text, Springer, 2018. https://link.springer.com/book/10.1007/978-3-319-73531-3
[7] McCallum, A., & Nigam, K., "A Comparison of Event Models for Naïve Bayes Text Classification," AAAI Workshop, 1998. https://www.cs.cmu.edu/~knigam/papers/text_classification.pdf
[8] Han, J., Kamber, M., & Pei, J., Data Mining: Concepts and Techniques, Elsevier, 2011. https://www.sciencedirect.com/book/9780123814791/data-mining-concepts-and-techniques
[9] Jurafsky, D., & Martin, J. H., Speech and Language Processing, Pearson, 2020. https://web.stanford.edu/~jurafsky/slp3/
[10] Maynez, J., Narayan, S., Bohnet, B., & McDonald, R., "On Faithfulness and Factuality in Abstractive Summarization," ACL, 2020. https://aclanthology.org/2020.acl-main.173/
[11] Ji, Z., Lee, N., Frieske, R., et al., "Survey of Hallucination in Natural Language Generation," ACM Computing Surveys, 2023. https://dl.acm.org/doi/10.1145/3571730
[12] Berners-Lee, T., "Information Management: A Proposal," CERN, 1989. https://www.w3.org/History/1989/proposal.html
[13] Marcus, G., "The Next Decade in Artificial Intelligence," *AI Magazine*, 2020. https://ojs.aaai.org/index.php/aimagazine/article/view/13917
[14] Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S., "On the Dangers of Stochastic Parrots," FAccT, 2021. https://dl.acm.org/doi/10.1145/3442188.3445922
[15] World Wide Web Consortium (W3C), "Web Architecture: Concepts and Design," 2019. https://www.w3.org/TR/webarch/



10.22214/IJRASET

INDEX COPERNICUS

45.98

IMPACT FACTOR: 7.129

TOGETHER WE REACH THE GOAL

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY