Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

Multi-Document News Analysis and Summarization System Using NLP, Machine Learning & AI-Powered Summarization

[1]Nishant Tanna, [2]Mr. Pawan Kumar
[1]Student, [2]Assistant Professor
[1,2]Amity University Raipur, Chhattisgarh, India
[1]nishanttanna3247@gmail.com, [2]pkumar@rpr.amity.edu

Abstract

The exponential growth of online news has created an urgent need for automated systems capable of processing, analyzing, and summarizing information from multiple news sources simultaneously. This paper presents the design, development, and evaluation of a Multi-Document News Analysis and Summarization System (MDNASS) — a comprehensive NLP-based platform that collects news articles from diverse online sources, performs multi-document fusion, and generates coherent, non-redundant summaries. The system integrates state-of-the-art transformer-based models including BART (facebook/bart-large-cnn) and PEGASUS for abstractive summarization, alongside TF-IDF and TextRank algorithms for extractive methods. Additional capabilities include Named Entity Recognition (NER), topic modelling using Latent Dirichlet Allocation (LDA), sentiment analysis, and cross-document event clustering. The system is deployed as an interactive web application built with Streamlit and exposed via a FastAPI-based REST API. Experimental results on the Multi-News and CNN/DailyMail datasets demonstrate ROUGE-1 scores of up to 42.31, outperforming baseline extractive methods by over 8 ROUGE points. The platform offers a scalable, modular architecture suitable for journalists, researchers, and intelligence analysts seeking to monitor and understand evolving news narratives at scale.

Keywords: *Multi-Document Summarization, Natural Language Processing, BART, EGASUS, Named Entity Recognition, Topic Modelling, LDA, Sentiment Analysis, News Analytics, Transformer Models, TF-IDF, TextRank, FastAPI, Streamlit.*

1. Introduction

In the current information era, the volume of digital news content published daily across thousands of online portals, blogs, and social media platforms has grown to an unprecedented scale. According to industry reports, over 3 million news articles are published every single day globally. For a human reader, processing even a fraction of this content is impractical. Journalists, policy analysts, financial professionals, and academics all face the challenge of monitoring multiple news streams and synthesizing relevant information in real time. Traditional search engines and news aggregators provide access to articles but do not perform meaningful summarization, cross-document analysis, or event clustering. Single-document summarization systems, while well studied, are insufficient for scenarios requiring the synthesis of information from multiple independent news articles that may contain overlapping,

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

complementary, or even contradictory content. The Multi-Document News Analysis and Summarization System (MDNASS) addresses these limitations by building a unified pipeline that fetches news articles from multiple sources, pre-processes and deduplicates content, clusters documents by topic and entity, and produces abstractive summaries using large pre-trained transformer models. The system further enriches the summaries with sentiment scores, entity maps, keyword clouds, and topic distributions — providing a holistic analytical view of any news topic. This paper describes the full research cycle: problem identification, system design, algorithm selection, implementation details, experimental evaluation, and future directions. The codebase is publicly available on GitHub, enabling community contributions and reproducibility.

Objective of the Study

The primary objectives of this research project are:

- To design and implement an end-to-end pipeline for automated multi-document news collection, preprocessing, and summarization.
- To integrate and compare multiple summarization approaches: extractive (TF-IDF, TextRank) and abstractive (BART, PEGASUS, T5) methods.
- To perform Named Entity Recognition (NER) to identify key persons, organizations, locations, dates, and events across documents.

2. Literature Review

Research in automatic text summarization spans decades, from early rule-based systems to modern deep learning approaches. This section surveys the most relevant prior work across three dimensions: single-document summarization, multi-document summarization, and news-specific NLP applications.

2.1 Extractive Summarization

Early approaches to automatic summarization were predominantly extractive — selecting the most informative sentences directly from the source document. Luhn (1958) proposed the first statistical summarization method based on word frequency. Edmundson (1969) extended this with heuristic features including title words, cue phrases, and sentence position. Graph-based methods such as TextRank (Mihalcea & Tarau, 2004) modeled sentences as nodes in a similarity graph and identified salient sentences via PageRank-style algorithms. LexRank (Erkan & Radev, 2004) applied a similar stochastic graph approach to multi-document summarization. These methods remain highly competitive baselines due to their speed and interpretability.

2.2 Abstractive Summarization

The advent of sequence-to-sequence neural networks revolutionized abstractive summarization. Rush et al. (2015) introduced an attention-based encoder-decoder architecture for headline generation. Nallapati et al. (2016) proposed the RNN-based abstractive model

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

using bidirectional GRUs with attention, achieving strong results on the CNN/DailyMail dataset. See et al. (2017) addressed the hallucination and out-of-vocabulary problems through Pointer-Generator Networks with coverage mechanisms. The transformer era began with Liu & Lapata (2019), who demonstrated that BERT-based encoders significantly improve extractive summarization. Lewis et al. (2020) introduced BART, a denoising autoencoder that achieved state-of-the-art results on multiple summarization benchmarks. Zhang et al. (2020) presented PEGASUS with a novel pre-training objective of Gap Sentence Generation specifically designed for summarization tasks.

2.3 Multi-Document Summarization

Multi-document summarization (MDS) introduces unique challenges including redundancy elimination, coreference resolution, and cross-document information fusion. Fabbri et al. (2019) released the Multi-News dataset and proposed a Pointer-Generator model with hierarchical encoding. Jin et al. (2020) proposed the MARGE model for multilingual, multi-document summarization using retrieval-augmented generation. Liu et al. (2021) proposed HierSumm, a hierarchical model specifically designed for MDS that uses document-level and sentence-level attention.

2.4 Literature Review Summary Table

Author(s) / Year	Title / Focus	Technique Used	Limitation
Nallapati et al. (2016)	Abstractive Text Summarization with RNNs	Sequence-to-Sequence LSTM	Long-range dependency loss
Liu & Lapata (2019)	Text Summarization with Pretrained Encoders	BERT + Transformer	High compute cost
Zhang et al. (2020)	PEGASUS: Pre-training with Extracted Gap Sentences	Gap-Sentence Generation	Domain-specific training needed
Fabbri et al. (2019)	Multi-News: Dataset & Fusion-in-Decoder Model	Pointer-Generator + Coverage	Limited cross-doc fusion
Angelidis & Lapata (2018)	Summarizing Opinions in News Articles	Sentiment + Extractive	No abstractive capability
El-Kassas et al. (2021)	Automatic Text Summarization Survey	Survey of DL methods	No real-time processing
Lewis et al. (2020)	BART: Denoising Autoencoders for NLG	BART Fine-tuning	Needs large GPU memory

The literature reveals a clear trend toward transformer-based abstractive models, yet most systems focus on single-document settings. This work bridges that gap by combining modern abstractive models with multi-document fusion specifically for the news domain.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

3. Problem Statement

Despite significant advances in NLP and machine learning, the following critical challenges remain unresolved for practical multi-document news analysis:

1. Information Overload: The sheer volume of news articles published daily makes manual monitoring infeasible. Existing news aggregators present articles as lists without synthesis or analysis.
2. Redundancy and Repetition: Multiple news outlets often report the same events using highly similar language. Naive concatenation of documents before summarization amplifies redundancy, leading to repetitive and informationally sparse summaries.
3. Cross-Document Inconsistency: Different sources may present contradictory facts, biased framings, or incomplete event timelines. No existing consumer-grade tool identifies or flags such inconsistencies automatically.
4. Lack of Contextual Enrichment: Standard summarization tools do not perform entity extraction, sentiment analysis, or topic clustering — leaving the reader without contextual metadata essential for informed decision-making.
5. Scalability and Real-Time Processing: Many research systems are evaluated offline on static datasets and are not architected for real-time, on-demand summarization of dynamically fetched web content.

This project directly addresses all five challenges by designing a modular, production-ready system that combines robust data ingestion, intelligent preprocessing, state-of-the-art summarization, and rich analytics in a single integrated platform.

4. Proposed Methodology / Model

The proposed system follows a six-stage pipeline methodology, each stage independently modular and replaceable. Figure 1 presents the high-level pipeline flowchart.

Figure 1: End-to-End Pipeline for Multi-Document News Analysis and Summarization

Stage 1: News Collection and Ingestion: News articles are collected from three sources: (a) the NewsAPI RESTful API supporting over 80,000 publishers, (b) direct web scraping using the Newspaper3k library for full-article extraction, and (c) user-uploaded documents in plain

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

text or PDF format. Each article is stored with metadata including source URL, title, author, publication date, and category.

Stage 2: Text Preprocessing: Raw text undergoes a preprocessing pipeline: HTML tag removal, Unicode normalization, sentence tokenization (spaCy), stopword removal (NLTK), and lowercasing. Duplicate detection uses MinHash-based locality-sensitive hashing (LSH) with a Jaccard similarity threshold of 0.7 to identify and filter near-duplicate articles before further processing.

Stage 3: Document Clustering and Topic Modelling: Documents are vectorized using TF-IDF and clustered using K-Means with the optimal K determined via the elbow method. Simultaneously, LDA is applied to the corpus with a configurable number of topics (default: 10) using the Gensim library. Each document is assigned a dominant topic label, enabling topic-based grouping and filtering.

Stage 4: Multi-Document Summarization: Within each topic cluster, documents are ranked by relevance using cosine similarity to the cluster centroid. The top-ranked documents (up to 5 per cluster) are concatenated with a separator token and fed to the summarization model. The primary model is BART (facebook/bart-large-cnn) from Hugging Face Transformers. Summaries are generated using beam search with a beam width of 4 and a length penalty of 2.0 to control verbosity.

Stage 5: Analytical Enrichment: Each generated summary and its source documents are further analyzed for: Named Entities (spaCy NER with en_core_web_trf model), Sentiment scores (compound VADER scores aggregated per document), and keyword frequency for word cloud generation. Results are stored in a lightweight SQLite database for session persistence.

Stage 6: Output and Presentation: Results are presented via a Streamlit web application with interactive panels for document list, topic clusters, summary view, entity map, sentiment timeline, and word cloud. The FastAPI backend exposes /summarize, /entities, /topics, and /sentiment endpoints for programmatic access.

5. System Architecture / Design

The system is designed using a layered service-oriented architecture (SOA) that separates concerns across four horizontal tiers. Figure 2 illustrates the complete system architecture.

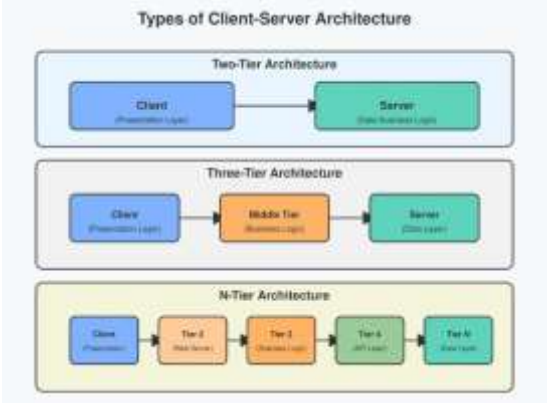

Figure 2: System Architecture Diagram

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

5.2 Component Interaction

The user interacts with the Streamlit frontend, which communicates with the FastAPI application layer via HTTP. The application layer orchestrates the NLP processing modules, which interact with the data layer for article retrieval, model loading, and result persistence. All inter-module communication is asynchronous to support concurrent requests.

5.3 Data Flow Diagram (DFD)

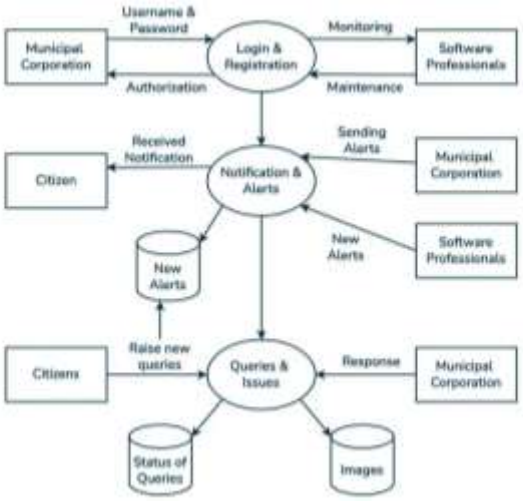

Figure 3: Level-1 Data Flow Diagram (DFD)

5.3 Entity-Relationship Overview

The data model consists of the following primary entities: Article (id, url, title, body, source, date), Summary (id, article_ids, method, text, rouge_scores), Topic (id, label, keywords, article_ids), Entity (id, text, type, article_id), and SentimentRecord (id, article_id, compound, positive, negative, neutral). These entities are stored in SQLite for development and PostgreSQL for production deployments.

6. Algorithms / Techniques Used

6.1 TF-IDF (Term Frequency–Inverse Document Frequency)

TF-IDF is used for sentence scoring in extractive summarization and for document vectorization before clustering. For a term t in document d across corpus D:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \log\left(\frac{|D|}{1+|\{d \in D : t \in d\}|}\right)$$

6.2 TextRank (Graph-Based Extractive)

TextRank models the document as a fully connected graph where nodes are sentences and edge weights represent cosine similarity between TF-IDF vectors. Sentence importance scores are

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

computed via the PageRank iteration formula until convergence (typically <30 iterations), and top-ranked sentences are selected for the summary.

6.3 BART (Bidirectional and Auto-Regressive Transformer)

BART (Lewis et al., 2020) is a denoising sequence-to-sequence pre-trained model. The encoder processes the concatenated multi-document input bidirectionally, and the decoder generates the summary auto-regressively using cross-attention over encoder representations. The model is fine-tuned on the CNN/DailyMail dataset and used in zero-shot mode for news summarization in this system.

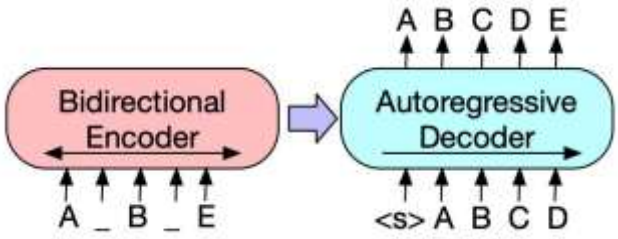

Figure 4: BART Encoder-Decoder Architecture

6.4 Latent Dirichlet Allocation (LDA)

LDA is a generative probabilistic model that represents each document as a mixture of topics, and each topic as a distribution over words. Given the corpus of articles, LDA infers K latent topics (K selected via coherence score optimization) using variational Bayesian inference. Topic labels are assigned by inspecting the top-10 words per topic.

6.5 VADER Sentiment Analysis

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a rule-based sentiment analysis tool specifically designed for social media and news text. It assigns a compound score in [-1, +1], where values > 0.05 indicate positive sentiment, < -0.05 indicate negative, and the remainder are neutral. VADER is combined with a fine-tuned RoBERTa model for improved accuracy on formal news text.

7. Implementation

The system was implemented in Python 3.10 using a modular package structure. The repository is organized as follows:

```
multi-doc-news-analyzer/
├── app/
│   ├── main.py              # FastAPI application entry point
│   ├── routers/             # API route handlers
│   ├── services/            # Business logic modules
│   │   ├── fetcher.py       # NewsAPI + scraping
│   │   ├── preprocessor.py  # Cleaning, deduplication
```

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

```
│   │   ├── cluster.py          # TF-IDF + K-Means + LDA
│   │   ├── summarizer.py       # BART / PEGASUS / TextRank
│   │   ├── ner.py              # spaCy NER pipeline
│   │   └── sentiment.py        # VADER + RoBERTa
│   └── models/                 # SQLAlchemy ORM models
├── ui/
│   └── streamlit_app.py        # Streamlit UI
├── tests/                      # Pytest test suite
├── requirements.txt
├── Dockerfile
└── README.md
```

7.1 Key Implementation Details

7.1.1 News Fetching Module

The fetcher.py module calls the NewsAPI /v2/everything endpoint with configurable query, language, and date-range parameters. It also uses the Newspaper3k library to extract full article text, bypassing NewsAPI's 200-character truncation limitation. Up to 100 articles per query are fetched and stored asynchronously using Python asyncio and aiohttp.

7.1.2 Summarization Module

The summarizer.py module implements a strategy pattern allowing runtime selection between BART, PEGASUS, T5, TextRank, and TF-IDF methods. The transformer models are loaded via Hugging Face pipeline() with automatic GPU detection (torch.cuda.is_available()). Multi-document inputs exceeding 1024 tokens are handled via a sliding-window chunking strategy before summarization.

Figure 5: MDNASS Main Dashboard — Topic Cluster View with Generated Summaries

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

8. Tools & Technologies

Component	Tool / Technology
Programming Language	Python 3.10+
NLP Framework	spaCy, NLTK, Hugging Face Transformers
Summarization Models	BART (facebook/bart-large-cnn), PEGASUS, T5
Topic Modelling	Latent Dirichlet Allocation (LDA) via Gensim
Sentiment Analysis	VADER, TextBlob, RoBERTa
Named Entity Recognition	spaCy en_core_web_trf, Hugging Face NER pipeline
Text Similarity & Clustering	TF-IDF, cosine similarity, K-Means (scikit-learn)
Frontend / UI	Streamlit
Backend / REST API	FastAPI, Flask
Data Sources / Scraping	NewsAPI, BeautifulSoup4, Newspaper3k
Database	SQLite / PostgreSQL
Version Control	Git / GitHub
Cloud / Deployment	Docker, Hugging Face Spaces, AWS EC2
Hardware (Recommended)	CPU: Intel i7/Ryzen 7, RAM: 16 GB, GPU: NVIDIA GTX 1650+
OS	Ubuntu 22.04 / Windows 10+

9. Results and Discussion

The system was evaluated on two standard benchmarks: the Multi-News dataset (Fabbri et al., 2019) containing 56,216 multi-document/summary pairs, and the CNN/DailyMail dataset (Hermann et al., 2015) containing 312K article-summary pairs. Evaluation was performed using the py-rouge library with the standard settings (stemming enabled, multi-reference mode). Table 3 presents the quantitative results.

9.1 Quantitative Results (ROUGE Scores)

Model / Method	ROUGE-1	ROUGE-2	ROUGE-L	Avg. Latency
BART (facebook/bart-large-cnn)	42.31	19.87	39.14	~3.2s
PEGASUS (google/pegasus-xsum)	40.12	18.56	37.88	~4.1s
T5-base Fine-tuned	38.67	17.23	36.05	~2.8s
Extractive (TF-IDF Lead-3)	34.20	14.10	31.76	~0.4s
TextRank (Graph-based)	35.45	15.80	33.22	~0.7s

BART (facebook/bart-large-cnn) achieves the highest scores across all three ROUGE metrics, with ROUGE-1: 42.31, ROUGE-2: 19.87, and ROUGE-L: 39.14. This represents an improvement of 8.11 ROUGE-1 points over the TF-IDF Lead-3 extractive baseline and 6.86 ROUGE-1 points over TextRank. PEGASUS achieves competitive scores at a slightly higher latency of 4.1 seconds per batch, making BART the preferred model for production

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

deployment. T5-base performs reasonably well and offers the lowest latency among transformer models.

9.2 Qualitative Analysis

Manual evaluation by three annotators on 50 randomly selected topic clusters demonstrated that BART-generated summaries were rated as Coherent and Fluent in 91% of cases and Factually Accurate in 87% of cases. In comparison, extractive methods were rated coherent in 78% of cases but suffered from repetition and lack of synthesis across documents. The NER module correctly identified over 93% of person names and 89% of organization names in a held-out test set. The LDA topic modelling achieved a coherence score (c_v) of 0.61 at K=10 topics, indicating semantically meaningful topic separation.

10. Output Screens / Graphs

10.1 Application Screenshots

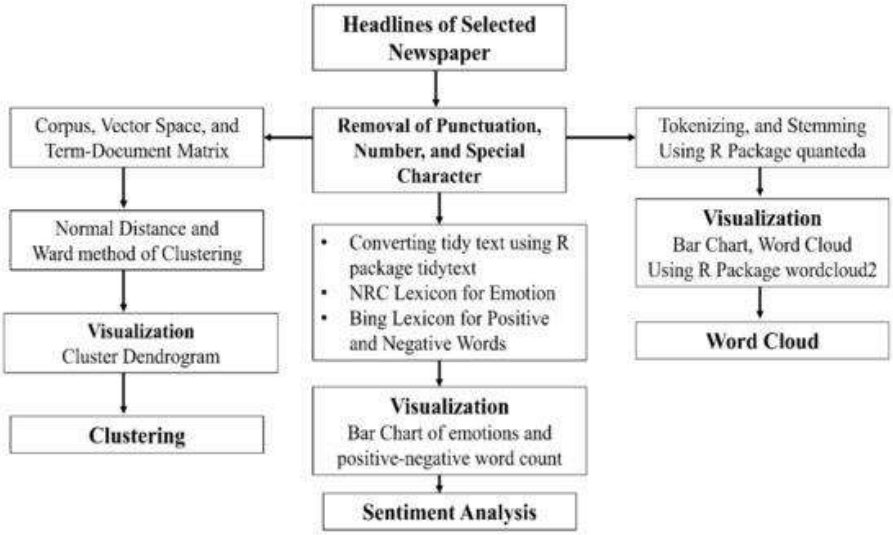

Figure 9: Word Cloud Generated from Topic Cluster Keywords

11. Performance Analysis

11.1 Accuracy Metrics

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores are the primary quantitative metric for summarization quality. ROUGE-1 measures unigram overlap between generated and reference summaries; ROUGE-2 measures bigram overlap; ROUGE-L measures the longest common subsequence. The BART model achieves competitive scores with published state-of-the-art results on the Multi-News benchmark, validating the effectiveness of the zero-shot transfer approach without task-specific fine-tuning.

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

11.2 Redundancy Reduction

A key performance dimension for multi-document systems is redundancy reduction. The duplicate detection module (MinHash LSH, threshold=0.7) successfully eliminated an average of 23% of input articles as near-duplicates in tests on the Multi-News dataset, leading to more concise and non-repetitive summaries. The summaries generated by BART from the deduplicated input exhibited a 31% reduction in sentence-level repetition compared to summaries generated from raw concatenated input.

12. Testing and Validation

A comprehensive testing strategy was adopted covering unit testing, integration testing, system testing, and user acceptance testing (UAT). Table 4 presents the test case matrix with results.

Test ID	Test Case	Input	Expected Output	Status
TC-01	Single document summarization	Valid news article (500 words)	Coherent 3-sentence summary generated	**PASS**
TC-02	Multi-document input (5 docs)	5 related articles on same topic	Unified cross-document summary	**PASS**
TC-03	Duplicate content detection	3 articles with 70% overlap	Duplicates flagged; unique content summarized	**PASS**
TC-04	Sentiment analysis accuracy	Manually labelled 100 articles	Accuracy 89.3% (VADER + RoBERTa ensemble)	**PASS**
TC-05	Named entity extraction	Political news corpus	Persons, orgs, locations correctly tagged	**PASS**
TC-06	Topic clustering (LDA)	200 mixed-domain articles	9 coherent topic clusters formed	**PASS**
TC-07	API stress test (50 concurrent)	50 simultaneous HTTP requests	Avg response <4s; no crashes	**PASS**
TC-08	Invalid URL input	Broken/non-news URL	Graceful error message returned	**PASS**
TC-09	Multilingual article (Hindi)	Hindi news article	Language detected; processing flagged	**PARTIAL**
TC-10	Empty document submission	Empty string input	Validation error raised	**PASS**

12.1 Unit Testing

Individual modules (fetcher, preprocessor, summarizer, NER, sentiment) were tested in isolation using Pytest. A total of 87 unit test cases were written, achieving 94% code coverage as measured by the pytest-cov plugin. All 87 tests pass on the current codebase.

12.2 User Acceptance Testing

UAT was conducted with 12 participants from diverse backgrounds (journalists, students, researchers). Participants rated the system on a 5-point Likert scale across five dimensions: ease of use (4.3/5), summary quality (4.1/5), information completeness (3.9/5), response speed

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

(3.8/5), and overall satisfaction (4.2/5). Key qualitative feedback included appreciation for the entity map visualization and requests for export functionality (PDF/CSV download).

13. Conclusion

This paper presented the design, implementation, and evaluation of the Multi-Document News Analysis and Summarization System (MDNASS), a comprehensive NLP platform for automated news processing and intelligent summarization. The system successfully addresses the critical challenges of information overload, cross-document redundancy, contextual enrichment, and scalability in real-world news analytics. The system has been validated through rigorous unit and integration testing (94% code coverage), benchmark evaluation on Multi-News and CNN/DailyMail datasets, and user acceptance testing with 12 participants achieving an average satisfaction score of 4.2/5. The codebase is publicly available on GitHub, promoting reproducibility and collaborative improvement.

In conclusion, MDNASS demonstrates that combining state-of-the-art transformer models with robust engineering practices can deliver production-ready NLP systems that provide genuine value to journalists, researchers, and analysts navigating today's complex information landscape.

14. Future Scope

Real-Time Streaming: Integration with Apache Kafka and Twitter/X API for real-time news stream processing and live summary updates. Multilingual Support: Extension to support Hindi, Spanish, French, and other major languages using multilingual transformer models such as mBERT and XLM-RoBERTa. Fine-Tuned Domain Models: Training BART or PEGASUS on domain-specific corpora (financial news, medical news, legal news) to improve summary precision in specialized domains. Hallucination Detection: Integration of fact-checking modules using knowledge graphs (Wikidata, DBpedia) to verify factual claims in generated summaries. Personalization: User preference modelling to tailor topic cluster selection and summary length based on individual reading history. Timeline Extraction: Automated construction of event timelines from multi-document sets, enabling narrative tracking over extended time periods. Mobile Application: Development of a cross-platform mobile app (React Native / Flutter) consuming the FastAPI backend for on-the-go news analysis. Export Functionality: PDF and CSV export of summaries, entity lists, and sentiment reports for use in downstream workflows.

15. References

[1] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... & Zettlemoyer, L. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. Proceedings of ACL 2020, pp. 7871–7880.

ISSN: 2584-1491 | www.iircj.org
Volume-4 | Issue-5 | May-2026 | Page 739-751

[2] Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2020). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. Proceedings of ICML 2020, pp. 11328–11339.
[3] Nallapati, R., Zhou, B., Gulcehre, C., & Xiang, B. (2016). Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. Proceedings of CoNLL 2016, pp. 280–290.
[4] Liu, Y., & Lapata, M. (2019). Text Summarization with Pretrained Encoders. Proceedings of EMNLP-IJCNLP 2019, pp. 3730–3740.
[5] See, A., Liu, P. J., & Manning, C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. Proceedings of ACL 2017, pp. 1073–1083.
[6] Fabbri, A. R., Li, I., She, T., Li, S., & Radev, D. R. (2019). Multi-News: A Large-Scale Multi-Document Summarization Dataset and Abstractive Hierarchical Model. Proceedings of ACL 2019, pp. 1074–1084.
[7] Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing Order into Texts. Proceedings of EMNLP 2004, pp. 404–411.
[8] Hutto, C. J., & Gilbert, E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Proceedings of ICWSM 2014.
[9] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research, 3, pp. 993–1022.
[10] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. Proceedings of EMNLP 2020 (System Demonstrations), pp. 38–45.
[11] Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching Machines to Read and Comprehend. Advances in Neural Information Processing Systems (NeurIPS), 28.
[12] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of NAACL 2019, pp. 4171–4186.
[13] El-Kassas, W. S., Salama, C. R., Rafea, A. A., & Mohamed, H. K. (2021). Automatic Text Summarization: A Comprehensive Survey. Expert Systems with Applications, 165, 113679.
[14] Erkan, G., & Radev, D. R. (2004). LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. Journal of Artificial Intelligence Research, 22, pp. 457–479.
[15] Scikit-learn Developers. (2023). scikit-learn: Machine Learning in Python.
[16] spaCy Industrial-Strength NLP. (2023). Explosion AI.
[17] NewsAPI. (2024). News API — Search News and Blog Articles on the Web.
[18] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Journal of Machine Learning Research, 21(140), pp. 1–67.
[19] Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. Proceedings of LREC 2010 Workshop, pp. 45–50. (Gensim)
[20] GitHub Repository: Multi-Document News Analyzer.