Comparative Analysis of Open-Source News Crawlers

rmfeeder Article

November 18, 2025

Comparative Analysis of Open-Source News Crawlers

Summary of Key Findings

This report evaluates six open-source news crawlers—**news-please**, **Fundus**, **news-crawler**, **news-crawl**, **Trafilatura**, and **newspaper4k**—focusing on extraction accuracy, supported sites, and ease of use. Fundus and Trafilatura lead in precision and recall for text extraction, while newspaper4k excels in multilingual support and NLP integration. News-please and news-crawl are optimized for large-scale archival, with trade-offs in speed and configurability. Below, we dissect each tool's strengths, weaknesses, and ideal use cases.

News-Please

Overview

news-please is a Python-based crawler designed for large-scale news extraction, integrating with CommonCrawl's archive for historical data retrieval[12].

Pros

- **CommonCrawl Integration**: Efficiently extracts articles from CommonCrawl's vast archive, ideal for longitudinal studies[12].
- **Structured Metadata**: Extracts titles, authors, publication dates, and multilingual content with 80+ language support[12].
- **Flexible Storage**: Supports JSON, PostgreSQL, Elasticsearch, and Redis[23].

Cons

- **Speed**: Slower processing (61x baseline in benchmarks) due to comprehensive metadata extraction[45].
- **IP Blocking**: Prone to throttling when scraping large sites like CNN[63].

- **Setup Complexity**: Requires manual configuration for Elasticsearch/Redis[23].

Conclusion: Best for researchers needing historical news data from CommonCrawl, but less suited for real-time scraping.

Fundus

Overview

Fundus uses **bespoke parsers** tailored to individual news sites, prioritizing extraction quality over quantity[789].

Pros

- **Highest Accuracy**: Achieves F1-scores of 97.69% in benchmarks, outperforming Trafilatura (93.62%) and news-please (93.39%)[685].
- **Structured Output**: Preserves article formatting (paragraphs, subheadings) and extracts meta-attributes like topics[78].
- **CommonCrawl Optimization**: Efficiently processes CC-NEWS datasets with multi-core support[82].

Cons

- **Limited Coverage**: Supports only predefined publishers (e.g., AP News, Reuters), restricting scalability[82].
- **Static Crawling**: Lacks real-time dynamic content handling[62].

Conclusion: Ideal for projects requiring artifact-free text from high-quality sources, but not for dynamic or unsupported sites.

News-Crawler (LuChang-CS)

Overview

A Python-based tool targeting major outlets like BBC and Reuters[10].

Pros

- **Ease of Use**: Simple CLI and Python API for small-scale scraping[10].
- **Versioning**: Tracks article changes over time, useful for longitudinal analysis[10].

Cons

- **Limited Benchmarking**: No public performance metrics compared to alternatives[10].
- **Resource-Intensive**: Struggles with large-scale crawls due to single-threaded design[10].

Conclusion: Suitable for academic projects with limited scope, but lacks enterprise-grade scalability.

News-Crawl (CommonCrawl)

Overview

A StormCrawler-based system producing WARC files for archival[11][10].

Pros

- **Archival Focus**: Generates WARC files compatible with CommonCrawl's AWS Open Dataset[11].
- **RSS/Sitemap Support**: Discovers articles via feeds, ensuring comprehensive coverage[11].

Cons

- **Complex Setup**: Requires Elasticsearch and Apache Storm, increasing deployment overhead[11].
- **No Content Extraction**: Stores raw HTML without text/metadata extraction[11].

Conclusion: Tailored for developers building news archives, not for direct content analysis.

Trafilatura

Overview

A Python/CLI tool optimized for precision and multilingual extraction[12135].

Pros

- **Benchmark Leader**: Outperforms Goose3, Boilerpipe, and Readability with 90.2% F1-score[135].
- **Lightweight**: Processes HTML 4.8x faster than news-please[125].
- **Metadata Retention**: Extracts publish dates, authors, and languages consistently[1314].

Cons

- **Speed vs. Recall**: Precision mode reduces recall by 3%[5].
- **Dynamic Content**: Struggles with JavaScript-rendered pages without Playwright integration[14].

Conclusion: The best all-rounder for most use cases, balancing speed, accuracy, and ease of use.

Newspaper4k

Overview

A revived fork of Newspaper3k with enhanced NLP and multithreading[615].

Pros

- **NLP Integration**: Generates summaries/extracts keywords, ideal for content curation[615].
- **Multithreading**: Downloads articles 15x faster than single-threaded tools[615].
- **Backward Compatibility**: Seamless migration from Newspaper3k[615].

Cons

- **Dependency Hell**: Requires manual installation of libxml2, Pillow, etc.[615].
- **Incomplete Fixes**: 180+ open GitHub issues, including inconsistent date parsing[615].

Conclusion: Optimal for developers needing NLP features and Google News scraping, despite setup hurdles.

Final Recommendations

By Use Case

1. **Highest Accuracy**: **Fundus** for academic/labelled datasets[78].
2. **General-Purpose**: **Trafilatura** for multilingual, precision-focused extraction[12514].
3. **NLP/Summarization**: **Newspaper4k** for keyword extraction and metadata[615].
4. **Historical Archives**: **news-please** or **news-crawl** for CommonCrawl integration[111].

Summary Table

Tool	Accuracy (F1)	Speed	Ease of Use	Best For
Fundus	97.69%[8]	Medium	Moderate	High-quality, predefined publishers
Trafilatura	90.2%[5]	High	High	Multilingual, general-purpose
Newspaper4k	94.6%[6]	High	Moderate	NLP features, Google News
news-please	85.81%[5]	Low	Low	CommonCrawl historical data

Note: Metrics derived from cited benchmarks.

Critical Considerations

- **Dynamic Content**: None of the tools natively handle JavaScript-heavy sites; pair with Playwright/Selenium[1415].

- **Legal Compliance**: Adhere to robots.txt and rate limits to avoid IP blocks[16][17].

By aligning tool capabilities with project requirements, users can optimize extraction quality and efficiency effectively[4][7][8][5].

⁂

Footnotes

1. https://github.com/fhamborg/news-please ↩ ↩2 ↩3 ↩4
2. https://github.com/free-news-api/news-crawlers ↩ ↩2 ↩3 ↩4 ↩5 ↩6 ↩7 ↩8
3. https://github.com/free-news-api/news-crawlers ↩ ↩2 ↩3
4. https://htmldate.readthedocs.io/en/latest/evaluation.html ↩ ↩2
5. https://trafilatura.readthedocs.io/en/latest/evaluation.html ↩ ↩2 ↩3 ↩4 ↩5 ↩6 ↩7 ↩8 ↩9 ↩10
6. https://github.com/free-news-api/news-crawlers ↩ ↩2 ↩3 ↩4 ↩5 ↩6 ↩7 ↩8 ↩9 ↩10 ↩11
7. https://arxiv.org/html/2403.15279 ↩ ↩2 ↩3 ↩4
8. https://aclanthology.org/2024.acl-demos.29.pdf ↩ ↩2 ↩3 ↩4 ↩5 ↩6 ↩7 ↩8
9. https://aclanthology.org/2024.acl-demos.29/ ↩
10. https://github.com/free-news-api/news-crawlers ↩ ↩2 ↩3 ↩4 ↩5 ↩6
11. https://github.com/commoncrawl/news-crawl ↩ ↩2 ↩3 ↩4 ↩5 ↩6
12. https://github.com/markusmobius/go-trafilatura ↩ ↩2 ↩3
13. https://trafilatura.readthedocs.io ↩ ↩2 ↩3

14. https://www.reddit.com/r/LangChain/comments/1ef12q6/the_rag_engineers_guide_to_document_parsing/ ↩ ↩2 ↩3 ↩4

15. https://www.reddit.com/r/Python/comments/1bmtdy0/i_forked_newspaper3k_fixed_bugs_and_improved_its/ ↩ ↩2 ↩3 ↩4 ↩5 ↩6 ↩7 ↩8

16. https://forage.ai/blog/introduction-to-news-crawling/ ↩

17. https://forage.ai/blog/introduction-to-news-crawling/ ↩