news-crawler-LM: A Small Long-Context Model For High-Quality News Crawling

Pascal Stolzenburg* **Jonas Golde*** **Max Dallabetta** **Alan Akbik**
Humboldt-Universität zu Berlin
pstolzenburg98@gmail.com

Abstract

Extracting structured content from news pages remains challenging due to heterogeneous HTML layouts, inconsistent markup, and substantial boilerplate such as navigation elements and advertisements. Rule-based news crawlers can achieve high extraction accuracy by encoding site-specific structure, but require manual configuration in order to generalize to new publishers. Large language models provide a more flexible alternative by reducing the need for handcrafted rules, but their high computational cost limits practical deployment. In this paper, we introduce news-crawler-LM, a small long-context language model fine-tuned on high-quality, human-validated extractions from the Fundus news-crawling library. Our model converts raw HTML into plaintext and structured JSON, including fields such as headline, author, publication date, and article body. In our experiments, news-crawler-LM outperforms strong baselines in HTML-to-Markdown and HTML-to-JSON extraction, improving performance by +4.8 BLEU and +6.1 METEOR in the HTML-to-Markdown task, and by +2.2 BLEU and +4.1 METEOR in the HTML-to-JSON task. However, we also observe that our model only slightly better compared to other rule-based parsing libraries on the HTML-to-plaintext task in evaluations on previously unseen publishers. We release all models and artifacts to the research community [1].

1 Introduction

Online news articles are a widely used data source for training and evaluating language models, particularly for tasks such as social and political analysis (Masud et al., 2020), information extraction (Yates et al., 2007; Whitehouse et al., 2023), or document-level reasoning such as fact checking (Mishra et al., 2020; Sathe et al., 2020). However, training language models on web-derived news data requires converting raw HTML pages into clean, structured representations (Hamborg et al., 2017). This preprocessing step remains challenging due to heterogeneous page layouts, inconsistent markup, and substantial boilerplate content such as navigation elements and advertisements, which can introduce noise and bias into downstream model training (Penedo et al., 2024).

	RULES	ML	OURS
Scalability	×	✓	✓
Extraction quality	✓	×	✓
Cross-domain	×	✓	✓
Transparency	✓	×	✓

Table 1: Comparison of news crawling approaches. news-crawler-LM (Ours) combines the strengths of rule-based (e.g., Fundus) and ML-based (e.g., LLMs) methods.

Current approaches for such conversion tasks show complementary limitations. First, rule-based extraction systems such as FUNDUS (Dallabetta et al., 2024) or Trafilatura (Barbaresi, 2021) encode explicit assumptions about website structure and can produce high-quality outputs, but require manual rule-engineering and frequent updates which makes it challenging to scale this approach to a large and evolving set of publishers. On the other end, large language models (LLMs) can be applied with minimal configuration and generalize across diverse HTML layouts, but their high computational cost and limited precision in structured extraction constrain their use for large-scale data preparation (Gur et al., 2023a; Shen et al., 2025). More fundamentally, we hypothesize the HTML-

*Equal contribution

[1] https://huggingface.co/stolzenp/FundusCrawler-plaintext, https://huggingface.co/stolzenp/FundusCrawler-json, https://huggingface.co/datasets/stolzenp/fundus-cleaned-filtered-62K

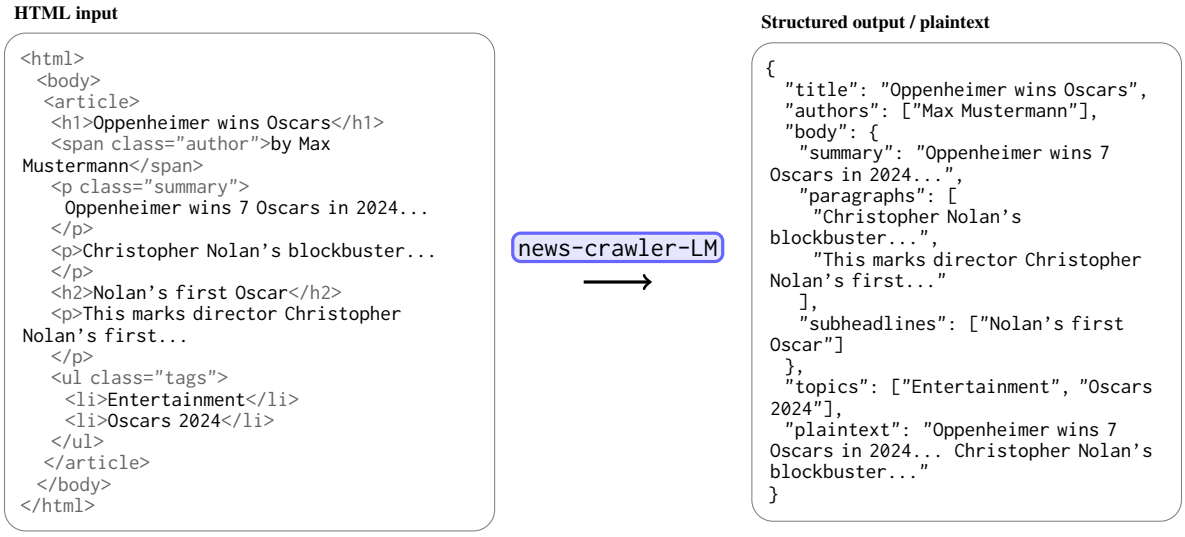

Figure 1: HTML conversion task: Given the raw HTML of a crawled news article, the model is trained to extract the article content (HTML-to-plaintext) and structured attributes, e.g. content, title, authors, topics (HTML-to-JSON).

conversion task may be most effective using small and efficient language models trained on low-noise, human-validated HTML-to-text pairs.

To this end, we present `news-crawler-LM`, a small long-context language model designed to bridge the gap between rule-based and generative extraction approaches. Our approach combines three components: *(i)* human-validated training data derived from the Fundus library, resulting in low-noise supervision; *(ii)* a strong Transformer backbone, namely ReaderLM-v2 (Wang et al., 2025), which is pre-trained for the HTML-to-text conversion task; and *(iii)* a contrastive training objective to reducing degeneration at inference time when processing long HTML sequences. Specifically, we train our model to perform two tasks: *(i)* HTML-to-plaintext, and *(ii)* HTML-to-JSON, containing fields such as title, author, publication date, and article body. In our evaluations, we find that `news-crawler-LM` outperforms competitive long-context baselines, including ReaderLM-v2 and Qwen2.5-32B. However, we also observe that performance is comparable to other parsing libraries beyond Fundus on the HTML-to-plaintext task, suggesting that for simpler settings, machine learning–based approaches may not be necessary to generalize to previously unseen publisher sites.

We summarize our contributions as follows:

1. We introduce `news-crawler-LM`, a small long-context language model for HTML-to-plaintext and HTML-to-JSON task for news articles.
2. We empirically evaluate `news-crawler-LM` with rule-based systems and strong long-context language models in zero-shot publisher settings.
3. We release model checkpoints, datasets, and training scripts to support reproducible research and practical deployment.[2]

2 `news-crawler-LM`

2.1 Dataset

Desiderata. Prior work on HTML conversion typically relies on large-scale crawling and heuristic filtering to construct training data. For example, ReaderLM-v2 obtains one million articles from CommonCrawl, where heuristics are applied to derive corresponding markdown representations. In contrast, we sample approximately 100k HTML-to-plaintext and JSON pairs extracted using human-defined rules from the Fundus library, and investigate whether this low-noise, high-quality supervision is sufficient to generalize across publishers.

Crawling Data From Fundus. We construct the training data $\mathcal{D}$ using the Fundus library (Dallabetta et al., 2024). Formally, we create the training data as a set of input–output pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, where each input x_i represents a cleaned HTML of a news article and each target y_i represents the extraction output. Depending on the task setting, y_i is either plaintext or a structured JSON serialization containing fields such as title, author, publication date, and article body.

[2]Removed for double-blinded review.

At the time of writing, we consider all 106 publishers supported by the Fundus library. For each publisher, we collect up to 1,000 articles to capture recurring publisher-specific HTML patterns. During crawling, 28 of the 106 publishers could not be fully retrieved due to rate limitations or changes in HTML structure. When possible, we supplement missing articles with older content from the CC-News dataset, which is natively supported by Fundus. After this process, we obtain a complete set of articles for 93 publishers, which constitute our core dataset spanning 25 languages.

Preprocessing and Filtering. We aim to support a context window of 32k tokens, reserving 24k tokens for the HTML input and 8k tokens for output generation. We filter irrelevant content such as JavaScript and <img>-tags following Wang et al. (2025), and discard documents whose HTML input exceeds the 24k token limit after filtering. This retains ∼79% of the data. None of the outputs (plaintext or JSON) exceeds the 8k token limit. We show the subword token distributions for HTML (input), plaintext, and JSON (targets) in Figure 2.

2.2 Model

Desiderata. General-purpose LLMs demonstrate strong zero-shot generalization and can handle HTML conversion without any task-specific training. However, their high computational cost makes them impractical for large-scale crawling pipelines. We therefore aim to distill this capability into a small, task-specialized model that approximates the behavior of human-written extraction rules while remaining efficient and scalable.

Backbone Model. We choose ReaderLM-v2 as our transformer backbone which has been continually pre-trained on HTML and markdown texts and post-trained using supervised fine-tuning (SFT) and direct preference optimization (DPO) (Rafailov et al., 2024) on HTML-to-markdown from randomly sampled CommonCrawl website[3]. ReaderLM-v2 builds upon Qwen2.5 (Qwen et al., 2025) and thus supports a context length of up to 32k tokens. Rather than aiming for diversity, we fine-tune the model on a small but high-quality dataset $\mathcal{D}$ using a standard language modeling objective, minimizing the negative log-likelihood for each training pair (x_i, y_i):

$$\mathcal{L} = -\sum_{t=1}^{T} \log p(y_{i,t} \mid y_{i,<t}, x_i).$$

2.3 Contrastive Learning For Isotopic Token Representations

Language models trained with a cross-entropy objective are known to exhibit text degeneration, particularly for long-context generation (Jiang et al., 2022). To mitigate this issue, we augment the standard language modeling objective with a contrastive loss following SimCTG (Su et al., 2022), which discourages overly similar token representations and reduces repetitive or unstable generation behavior.

Specifically, let $x = (x_1, \ldots, x_{|x|})$ denote the input sequence with corresponding hidden representations $(h_{x_1}, \ldots, h_{x_{|x|}})$. The contrastive term is defined using cosine similarity and a margin $\rho \in [-1, 1]$:

$$f(x_i, x_j) = \max(0,\ \rho - s(h_{x_i}, h_{x_i}) + s(h_{x_i}, h_{x_j}))$$

with

$$s(h_{x_i}, h_{x_j}) = \frac{h_{x_i}^{\top} h_{x_j}}{\|h_{x_i}\|\,\|h_{x_j}\|}.$$

Thus, the overall contrastive loss becomes:

$$\mathcal{L}_{\text{CL}} = \frac{1}{|x|(|x|-1)} \sum_{i=1}^{|x|} \sum_{\substack{j=1 \\ j \neq i}}^{|x|} f(x_i, x_j),$$

encouraging isotropic token representations. Following Su et al. (2022), we set $\rho = 0.5$. Our final training objective is:

$$\mathcal{L} = \mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{CL}}.$$

3 Experimental Setup

3.1 Data

We split the dataset by publisher into training, validation, and test sets, ensuring that no publisher appears in more than one split. We reserve 7 publishers for validation (∼7.7k samples) and sample 100 HTML inputs from each of 10 held-out publishers for testing, with the remaining data used for training.

We show the distribution of languages in the overall dataset in Table 2 and observe that the majority of articles are in Latin script. However, there are also publishers using Japanese or Hindi script.

Further, we show the distribution of subword token counts for HTML inputs, plaintext and JSON outputs in Figure 2.

[3] https://commoncrawl.org/blog/common-crawl-url-index

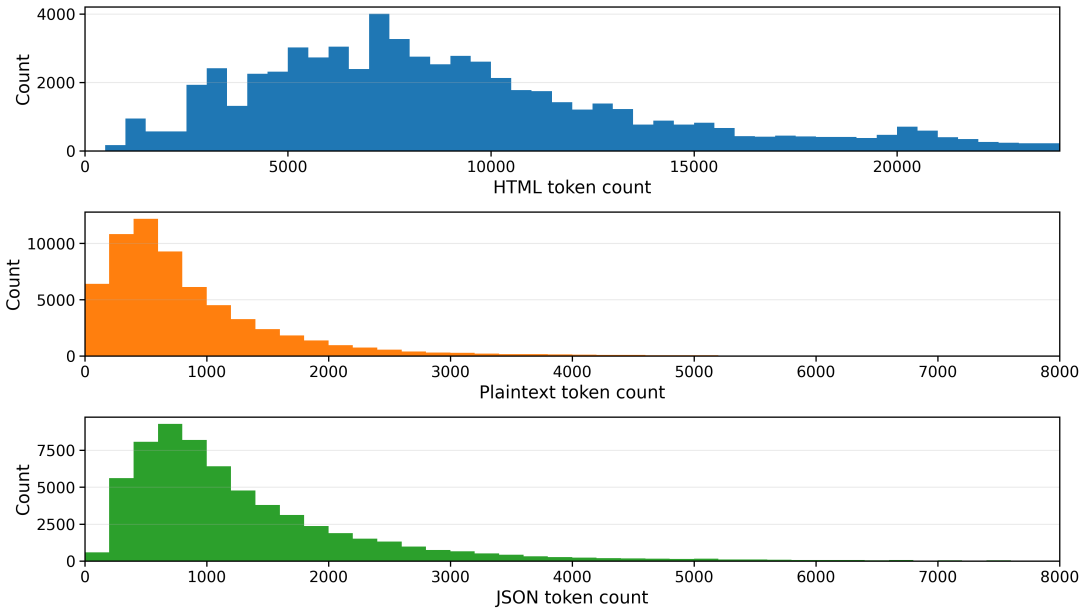

Figure 2: Distributions of subword token counts of the training data. We show HTML input counts (top), plaintext output counts (middle), and JSON output counts (bottom).

LANGUAGE	SHARE (%)
German	50.4
English	36.5
French	3.2
Norwegian	3.2
Turkish	2.2
Spanish	1.2
Hindi	1.1
Japanese	1.1
Others	1.1

Table 2: Language Distribution of Crawled Articles

3.2 Training

We train the model using 8-bit Adam (Kingma and Ba, 2017; Dettmers et al., 2022) with a batch size of 1, a learning rate of $4e^{-5}$, and a weight decay of 0.01. Training runs for 60k steps with a linear warmup over the first 1k steps, after which the learning rate decays linearly to zero. We select the best checkpoint based on validation perplexity with early stopping (patience = 3), evaluating every 1k steps. For parameter-efficient fine-tuning, we employ LoRA (Hu et al., 2021) with rank $r = 16$, scaling factor $\alpha = 16$, and no dropout, applying adapters to the attention projections (q, k, v, o) as well as the gating and feed-forward layers. This configuration keeps the number of trainable parameters small while allowing the model to adapt effectively to the HTML conversion task.

3.3 Metrics

BLEU (Papineni et al., 2002) measures n-gram precision between the predicted and reference text, combined with a brevity penalty to discourage overly short outputs. We compute BLEU using up to 4-grams.

METEOR (Banerjee and Lavie, 2005) aligns unigrams between prediction and reference, supporting exact, stemmed, and synonymous matches. Unlike BLEU, it accounts for word order by penalizing fragmented alignments. We use the default parameterization of Banerjee and Lavie (2005).

ROUGE-L (Lin, 2004) measures overlap via the longest common subsequence (LCS) between prediction and reference, capturing in-order matches without requiring contiguous spans.

Levenshtein Distance (Levenshtein, 1965) counts the minimum number of character-level edit operations (insertions, deletions, substitutions) needed to transform the prediction into the reference. We report the normalized distance to account for varying sequence lengths.

Jaro-Winkler Similarity (Winkler, 1990) extends the Jaro metric by adding a prefix-based weighting that boosts similarity scores when strings share a common prefix. This makes it particularly suited for structured fields such as titles or author names, where early character agreement is especially informative.

MODEL	BLEU ↑	METEOR ↑	ROUGE-L ↑	LEV. ↓	JARO-W. ↑
Parsing libraries					
news-please	0.599	0.702	0.669	0.386	0.741
Boilerpipe	0.599	0.654	0.680	0.380	0.728
Pre-trained language models					
ReaderLM-v2	0.098	0.294	0.275	0.830	0.544
Qwen2.5-32B-Instruct	0.077	0.249	0.168	0.821	0.601
Fine-tuned language models					
Qwen2.5-1.5B	0.590	0.652	0.677	0.384	0.774
Qwen2.5-1.5B + SimCTG	0.623	0.678	0.707	0.342	0.786
`news-crawler-LM`	**0.671**	**0.739**	**0.749**	**0.295**	**0.820**

Table 3: Results for HTML-to-plaintext task.

3.4 Baselines

We compare `news-crawler-LM` against three publicly available rule-based libraries that rely on human-crafted heuristics or HTML DOM parsing: Trafilatura (Barbaresi, 2021), news-please (Hamborg et al., 2017), and Boilerpipe (Kohlschütter et al., 2010). As model-based baselines, we include ReaderLM-v2 (Wang et al., 2025), Qwen2.5-1.5B, and Qwen2.5-32B-Instruct (Qwen et al., 2025).

A limitation of our evaluation is that the reference outputs are derived from the Fundus extraction rules, which introduces a potential bias in favor of `news-crawler-LM`, as it is trained to reproduce these very decisions. Other parsing libraries may follow different design philosophies regarding which content to include or exclude, and thus their outputs may diverge from the reference not due to lower quality, but due to differing extraction objectives. This should be taken into account when interpreting overlap-based metrics such as BLEU or ROUGE-L.

4 Results

4.1 HTML-to-Plaintext

Table 3 shows the performance of all methods on the HTML-to-plaintext task. Among all fine-tuned approaches, `news-crawler-LM` achieves the strongest performance across all metrics, outperforming both rule-based parsing libraries and other model-based baselines.

Comparing against rule-based libraries, `news-crawler-LM` consistently outperforms both news-please and Boilerpipe across all reported metrics, demonstrating that task-specific fine-tuning on high-quality supervision can surpass hand-crafted heuristics for structured news extraction.

Among model-based approaches, we observe a clear performance gap between pre-trained and fine-tuned models. Pre-trained models, both ReaderLM-v2 and Qwen2.5-32B-Instruct, perform substantially worse despite their scale, indicating that zero-shot generalization is insufficient for accurate HTML-to-plaintext extraction. Fine-tuning drastically closes this gap, with even the smallest fine-tuned model, Qwen2.5-1.5B, achieving competitive performance with the rule-based libraries. Furthermore, the addition of SimCTG training yields consistent improvements over standard fine-tuning across all metrics, confirming the benefit of contrastive objectives for structured text generation. Finally, `news-crawler-LM` outperforms Qwen2.5-1.5B across all measures, suggesting that continued pretraining on HTML data, as provided by the ReaderLM backbone, further improves performance on structured news parsing beyond standard fine-tuning alone.

4.2 HTML-to-JSON

Table 4 presents results on the HTML-to-JSON task. Consistent with the plaintext results, pre-trained models without fine-tuning perform poorly across all metrics, with BLEU, METEOR, and ROUGE-L scores below 0.24 and high Levenshtein distances for both ReaderLM-v2 and Qwen2.5-32B-Instruct. This confirms that zero-shot application of general-purpose language models is insufficient for reliable structured metadata extraction.

Task-specific fine-tuning again yields substantial improvements across all metrics. `news-crawler-LM` achieves the best overall performance on overlap-based metrics, outper-

MODEL	BLEU ↑	METEOR ↑	ROUGE-L ↑	LEV. ↓	JARO-W. ↑
Pre-trained language models					
ReaderLM-v2	0.059	0.213	0.219	0.871	0.487
Qwen2.5-32B-Instruct	0.073	0.236	0.165	0.821	0.578
Fine-tuned language models					
Qwen2.5-1.5B	0.309	0.466	0.598	0.448	0.713
Qwen2.5-1.5B + SimCTG	0.276	0.434	0.552	0.508	0.690
`news-crawler-LM`	**0.331**	**0.507**	**0.636**	**0.426**	**0.729**

Table 4: Results for HTML-to-JSON task.

MODEL	F1 (ALL)	F1 (VALID JSON)	VALID JSON ↑
Pre-trained language models			
ReaderLM-v2	0.008	0.025	0.200
Qwen2.5-32B-Instruct	0.006	0.020	0.144
Fine-tuned language models			
Qwen2.5-1.5B	**0.595**	**0.786**	0.644
Qwen2.5-1.5B + SimCTG	0.542	0.713	**0.656**
`news-crawler-LM`	0.523	0.782	0.533

Table 5: JSON extraction performance. F1 (ALL) is computed over all outputs, counting invalid JSON as empty predictions. F1 (VALID JSON) is computed only on syntactically valid JSON outputs. VALID JSON (%) denotes the fraction of generations that could be parsed as valid JSON.

forming all baselines on every reported measure. Notably, it improves over Qwen2.5-1.5B by +0.022 BLEU, +0.041 METEOR, and +0.038 ROUGE-L, while also reducing Levenshtein distance by 0.022. Interestingly, adding SimCTG training to Qwen2.5-1.5B does not yield the same gains as observed in the plaintext task, suggesting that the contrastive objective may be less beneficial for structured JSON generation.

To complement the overlap-based evaluation, we further assess syntactic validity and attribute-level accuracy in Table 5, reporting F1 scores for key generation alongside the fraction of syntactically valid outputs. Among fine-tuned models, Qwen2.5-1.5B attains the highest F1 both over all outputs (0.595) and when restricted to valid JSON (0.786), while maintaining a valid JSON rate of 64.44%. The SimCTG variant achieves the highest syntactic validity (65.56%) but lower F1 scores, suggesting a trade-off between structural correctness and semantic accuracy. `news-crawler-LM` achieves a lower valid JSON rate (53.33%) and F1 over all outputs (0.523), but its F1 restricted to valid outputs (0.782) is on par with Qwen2.5-1.5B, indicating that when `news-crawler-LM` does produce valid JSON, the extraction quality is competitive. Overall, these results confirm that task-specific fine-tuning is essential for effective HTML-to-JSON extraction, and that continued pretraining on HTML data supports competitive semantic accuracy even under stricter structural constraints.

Finally, rule-based parsing libraries are excluded from this evaluation, as they do not produce structured metadata in the format defined by Fundus.

5 Ablations

5.1 Repetition and Degeneration Analysis

Text degeneration in the form of repetitive outputs is a known failure mode of autoregressive language models (Holtzman et al., 2020). We analyze this tendency by measuring 5-gram repetition in the generated outputs, computing the fraction of outputs in which any 5-gram exceeds a repetition threshold of $t = 0.1$, i.e., appears in more than 10% of tokens. A higher low-repetition rate thus indicates more fluent and diverse generation. Results are shown in Table 6.

The pre-trained ReaderLM-v2 exhibits a very low low-repetition rate (0.078), indicating severe degeneration in the absence of task-

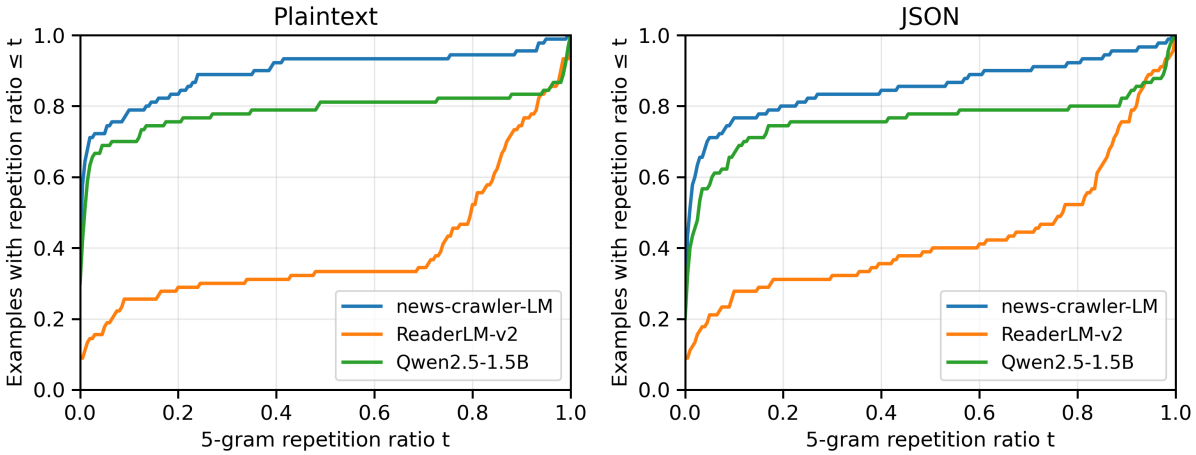

Figure 3: Cumulative distribution of 5-gram repetition ratios for plaintext (left) and JSON (right) generation. Each curve shows the fraction of outputs whose repetition ratio falls below threshold t. A curve shifted to the upper left indicates less repetitive generation. `news-crawler-LM` consistently produces the least repetitive outputs across both tasks.

MODEL	LOW-REP. RATE
Pre-trained	
ReaderLM-v2	0.078
Fine-tuned	
Qwen2.5-1.5B	0.667
`news-crawler-LM`	**0.767**

Table 6: Fraction of outputs with a 5-gram repetition ratio below $t = 0.1$.

specific fine-tuning. Among fine-tuned models, `news-crawler-LM` achieves the highest low-repetition rate (0.767), outperforming Qwen2.5-1.5B by 10 absolute points. This suggests that the SimCTG-based contrastive training objective, which explicitly encourages isotropic token representations, is effective at reducing repetitive generation in the context of structured HTML extraction.

To further characterize the distribution of repetitive outputs, Figure 3 plots the cumulative fraction of examples with a 5-gram repetition ratio below threshold t for both tasks. Across both plaintext and JSON generation, `news-crawler-LM` consistently dominates the cumulative distribution, indicating that a larger fraction of its outputs exhibit low repetition at any given threshold. ReaderLM-v2 shows a markedly different profile, with its distribution rising slowly and remaining flat over a wide range of t, suggesting that a substantial portion of its outputs suffer from severe repetition. Qwen2.5-1.5B in between FundusCrawler and ReaderLM, performing comparably to `news-crawler-LM` at very low thresholds but falling behind as t increases. These results are consistent across both tasks and corroborate the quantitative findings in Table 6, confirming that the SimCTG-based training objective leads to more fluent and less degenerate outputs.

5.2 Output Quality Distribution Analysis.

Figure 4 presents the complementary cumulative distribution of ROUGE-L scores across all test examples for both tasks. `news-crawler-LM` consistently dominates the distribution in both plaintext and JSON generation, meaning that a larger fraction of its outputs exceed any given ROUGE-L threshold k. For instance, in the plaintext task, `news-crawler-LM` maintains above 80% of outputs with ROUGE-L ≥ 0.4, whereas Qwen2.5-1.5B drops below this fraction considerably earlier. ReaderLM-v2, which is not fine-tuned, degrades rapidly and approaches zero well before $k = 1.0$, confirming that task-specific fine-tuning is critical for reliable extraction quality. Notably, the gap between `news-crawler-LM` and Qwen2.5-1.5B is more pronounced in the plaintext task than in JSON generation, where the two fine-tuned models perform more similarly across most of the distribution.

6 Related Work

Early work on web content extraction relied on heuristic and rule-based processing of HTML structure, often using DOM tree analysis to separate content from boilerplate (Gupta et al., 2005). While

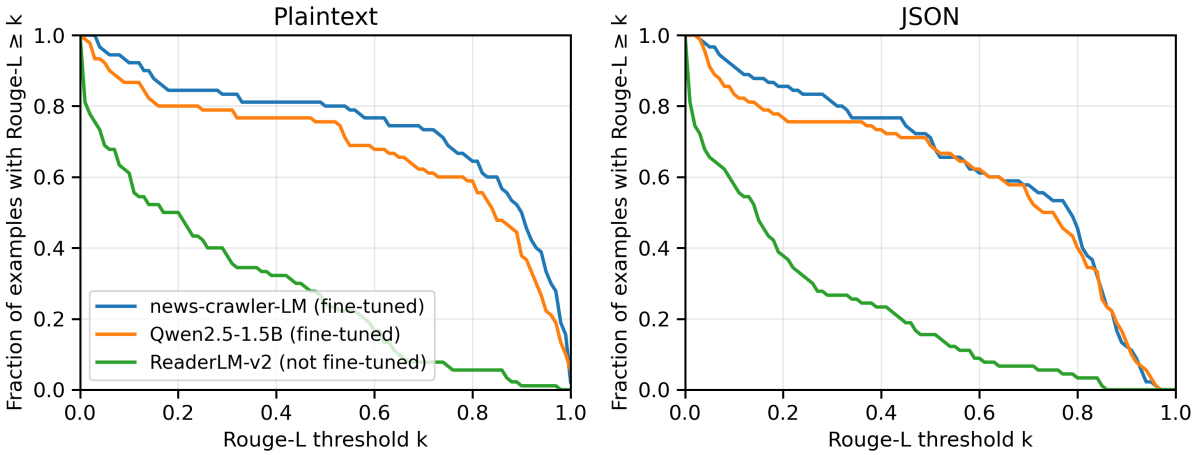

Figure 4: Complementary cumulative distribution of ROUGE-L scores for plaintext (left) and JSON (right) generation. Each curve shows the fraction of outputs achieving a ROUGE-L score of at least k. A curve shifted to the upper right indicates higher overall extraction quality. `news-crawler-LM` consistently achieves the highest ROUGE-L scores across both tasks.

effective for stable websites, these approaches degrade on heterogeneous layouts, motivating later learning-based methods that jointly model DOM structure and text to improve robustness and generalization (Lin et al., 2020; Deng et al., 2022).

Heuristic / Rule-Based Approaches. Recent line of works in forms of content extraction libraries transform web pages into text without relying on LLMs, typically combining heuristics and human-crafted rules to remove boilerplate content (Barbaresi, 2021; Hamborg et al., 2017; Pomikálek et al., 2012; Kohlschütter et al., 2010; Finn et al., 2001; Leonhardt et al., 2020). While broadly applicable, such generic extractors do not generalize well due to publisher-specific markup and achieve lower extraction precision. For example, Fundus (Dallabetta et al., 2024) relies on manually curated, publisher-specific rules, enabling high extraction accuracy but requiring substantial effort to adapt to new publishers. Our work builds on these developments by using rule-based extractions as low-noise supervision, enabling language models to generalize to new publishers.

Machine Learning-Based Approaches. On the other hand, LLMs have been explored as universal HTML-to-text extractors, converting HTML to markup (Gur et al., 2023b). Prior work shows that explicitly modeling HTML structure and additional signals such as layout or visual cues can improve extraction performance (Tan et al., 2025; Jung et al., 2022), and LLMs have been applied to large-scale document preprocessing (Xu et al., 2024) or interactive extraction settings (Bohra et al., 2025). However, general-purpose LLMs often have billions of parameters, motivating our work on specialized long-context models for the HTML-to-text task (Kim et al., 2025; Wang et al., 2025).

7 Conclusion

We introduced `news-crawler-LM`, a small, domain-specialized language model that converts raw HTML from news websites into plaintext or structured JSON. Our approach utilizes rule-based extractions from the Fundus library as supervision signal, builds on the domain-adapted ReaderLM-v2 backbone, and incorporates an additional contrastive training objective to improve output stability and reduce degeneration.

Our evaluation shows that `news-crawler-LM` learns consistent parsing behavior and outperforms large general-purpose LLMs across all considered metrics, while requiring substantially fewer computational resources. The model also surpasses similarly sized language models fine-tuned on the same dataset, demonstrating the importance of the additional contrastive objective. Moreover, our quality distribution analysis indicates that the majority of outputs achieve high ROUGE-L scores, with only less than 20% having a ROUGE-L scores < 0.6.

Overall, our results show that targeted fine-tuning provides a practical alternative to manual rule engineering and computationally intensive large-scale LLM inference for web article extraction.

Limitations

Although `news-crawler-LM` achieves strong performance, several limitations remain. First, the approach relies on high-quality rule-based supervision from Fundus, which may introduce biases or omissions present in the source extraction logic. While the models generalize to unseen publishers, they may still fail when confronted with radically different markup conventions, dynamically rendered content, or paywalled layouts.

Second, although contrastive learning reduces degeneration, a non-negligible portion of outputs still contain malformed JSON structures or missing fields, indicating the need for format-constrained decoding or post-hoc validation.

Third, the current work focuses primarily on English and German publishers, and generalization to low-resource languages and multilingual mixed-layout pages remains an open challenge.

Fourth, inference on very long documents may be bottleneck using classical models using self-attention. In our observations, generating an extraction for a single long HTML document without specific inference optimizations can take several minutes. Although this remains more efficient (in terms of compute) than very large LLM-based approaches, it may not meet latency requirements in high-throughput production settings. Addressing these limitations requires further exploration of models using mixture-of-experts (Cai et al., 2025) of linear attention (Katharopoulos et al., 2020), in which the computational complexity can be reduced.

Finally, as we train generative models to parse HTML into structured outputs, our approach is inherently susceptible to hallucinations. In some cases, the model may introduce information that is not explicitly supported by the input document, for example by inferring missing fields or generating plausible but incorrect metadata. While such behavior is common in sequence generation models, it poses particular challenges for information extraction tasks, where correctness and faithfulness to the source are critical. Mitigating hallucinations may require stronger input grounding, constrained decoding strategies, or explicit verification mechanisms.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Adrien Barbaresi. 2021. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 122–131, Online. Association for Computational Linguistics.

Arth Bohra, Manvel Saroyan, Danil Melkozerov, Vahe Karufanyan, Gabriel Maher, Pascal Weinberger, Artem Harutyunyan, and Giovanni Campagna. 2025. Weblists: Extracting structured information from complex interactive websites using executable llm agents. *Preprint*, arXiv:2504.12682.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*, page 1–20.

Max Dallabetta, Conrad Dobberstein, Adrian Breiding, and Alan Akbik. 2024. Fundus: A simple-to-use news scraper optimized for high quality extractions. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 305–314, Bangkok, Thailand. Association for Computational Linguistics.

Xiang Deng, Prashant Shiralkar, Colin Lockard, Binxuan Huang, and Huan Sun. 2022. Dom-lm: Learning generalizable representations for html documents. *Preprint*, arXiv:2201.10608.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. *9th International Conference on Learning Representations, ICLR*.

Aidan Finn, Nicholas Kushmerick, and Barry Smyth. 2001. Fact or fiction: Content classification for digital libraries. In *DELOS Workshops / Conferences*.

Suhit Gupta, Gail E. Kaiser, Peter Grimm, Michael F. Chiang, and Justin Starren. 2005. Automating content extraction of html documents. *World Wide Web*, 8(2):179–224.

Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. 2023a. Understanding HTML with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2803–2821, Singapore. Association for Computational Linguistics.

Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. 2023b. Understanding html with large language models. *Preprint*, arXiv:2210.03945.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*, pages 218–223.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *Preprint*, arXiv:1904.09751.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Shaojie Jiang, Ruqing Zhang, Svitlana Vakulenko, and Maarten de Rijke. 2022. A simple contrastive learning objective for alleviating neural text degeneration. *Preprint*, arXiv:2205.02517.

Geunseong Jung, Sungjae Han, Hansung Kim, Kwanguk Kim, and Jaehyuk Cha. 2022. Don't read, just look: Main content extraction from web pages using visual features. *Preprint*, arXiv:2110.14164.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. *Preprint*, arXiv:2006.16236.

Soyeon Kim, Namhee Kim, and Yeonwoo Jeong. 2025. Next-eval: Next evaluation of traditional and llm web data record extraction. *Preprint*, arXiv:2505.17125.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, page 441–450, New York, NY, USA. Association for Computing Machinery.

Jurek Leonhardt, Avishek Anand, and Megha Khosla. 2020. Boilerplate removal using a neural sequence labeling model. In *Companion Proceedings of the Web Conference 2020*, WWW '20, page 226–229, New York, NY, USA. Association for Computing Machinery.

Vladimir I. Levenshtein. 1965. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710.

Bill Yuchen Lin, Ying Sheng, Nguyen Vo, and Sandeep Tata. 2020. Freedom: A transferable neural architecture for structured information extraction on web documents. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 1092–1102. ACM.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Sarah Masud, Subhabrata Dutta, Sakshi Makkar, Chhavi Jain, Vikram Goyal, Amitava Das, and Tanmoy Chakraborty. 2020. Hate is the new infodemic: A topic-aware modeling of hate speech diffusion on twitter. *Preprint*, arXiv:2010.04377.

Rahul Mishra, Dhruv Gupta, and Markus Leippold. 2020. Generating fact checking summaries for web claims. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 81–90, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *Preprint*, arXiv:2406.17557.

Jan Pomikálek, Miloš Jakubíček, and Pavel Rychlý. 2012. Building a 70 billion word corpus of English from ClueWeb. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 502–506, Istanbul, Turkey. European Language Resources Association (ELRA).

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Aalok Sathe, Salar Ather, Tuan Manh Le, Nathan Perry, and Joonsuk Park. 2020. Automated fact-checking of claims from Wikipedia. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6874–6882, Marseille, France. European Language Resources Association.

Xingyu Shen, Shengding Hu, Xinrong Zhang, Xu Han, Xiaojun Meng, Jiansheng Wei, Zhiyuan Liu, and Maosong Sun. 2025. AutoClean: LLMs can prepare

their training corpus. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 85–95, Albuquerque, New Mexico. Association for Computational Linguistics.

Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. 2022. A contrastive framework for neural text generation. *Preprint*, arXiv:2202.06417.

Jiejun Tan, Zhicheng Dou, Wen Wang, Mang Wang, Weipeng Chen, and Ji-Rong Wen. 2025. Htmlrag: Html is better than plain text for modeling retrieved knowledge in rag systems. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, page 1733–1746. ACM.

Feng Wang, Zesheng Shi, Bo Wang, Nan Wang, and Han Xiao. 2025. Readerlm-v2: Small language model for html to markdown and json. *Preprint*, arXiv:2503.01151.

Chenxi Whitehouse, Clara Vania, Alham Fikri Aji, Christos Christodoulopoulos, and Andrea Pierleoni. 2023. WebIE: Faithful and robust information extraction on the web. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7734–7755, Toronto, Canada. Association for Computational Linguistics.

William E Winkler. 1990. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage.

Zhipeng Xu, Zhenghao Liu, Yukun Yan, Zhiyuan Liu, Ge Yu, and Chenyan Xiong. 2024. Cleaner pretraining corpus curation with neural web scraping. *Preprint*, arXiv:2402.14652.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael Cafarella, Oren Etzioni, and Stephen Soderland. 2007. TextRunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26, Rochester, New York, USA. Association for Computational Linguistics.