

LongListBench: A Benchmark for Long-List Entity Extraction Under Layout and OCR Noise

Serhii Shchoholiev^{1*} Anton Fedoruk^{1†} Akhil Mehta^{1‡}

¹Kay.ai, Brooklyn, NY, USA

January 21, 2026

Abstract

Existing datasets for evaluating document extraction using large language models (LLMs) predominantly focus on key-value pair extraction and fail to address the complexities of long-list entity extraction. However, semi-structured business documents with long incident/line-item lists—such as invoices, purchase orders, and claims summaries—commonly organize data in table-like structures with dozens to hundreds of repetitive entities of the same type, creating a critical gap in current benchmarking efforts. We introduce LongListBench, a synthetic benchmark for long-list entity extraction in semi-structured business documents, inspired by recurring patterns observed in real-world claims documents. Each instance pairs structured ground truth (JSON) with a rendered PDF and an OCR transcript produced by a vision-language model, enabling reproducible evaluation of end-to-end pipelines under layout and transcription noise. We systematically inject seven document phenomena observed in production settings, including page breaks, multi-row entities, exact duplicates, large documents, multiple tables, multi-column layout, and merged cells. Across 80 documents (6,828 incident rows), identifier coverage in OCR is near-perfect, while schema-conformant zero-shot LLM baselines achieve 81.9%, 80.0%, and 78.1% average field-level F1 for Gemini 2.5, GPT-4o, and GPT-5.2, respectively, averaged across the full benchmark. LongListBench targets the measurement problem; algorithmic solutions for robust long-list extraction are left to future work.

1 Introduction

Long-list entity extraction—recovering dozens to hundreds of repeated records from semi-structured documents—is a core requirement for document automation in domains such as insurance, finance, and procurement. While recent advances in document understanding

*serhii@kay.ai

†anton@kay.ai

‡akhil@kay.ai

models (e.g., layout-aware pretraining [1] and OCR-free approaches [2]) and general-purpose LLMs have improved extraction quality, robust evaluation of long-list scenarios remains limited.

Many established benchmarks focus on key-value style extraction or relatively short, form-like documents (e.g., FUNSD [3]) or narrow document types such as receipts (SROIE [4]). More recent datasets such as DocILE [5] include business documents and line items, but long lists in the wild often exhibit additional failure modes: repeated entities, page breaks, multi-column reading order, irrelevant tables, and table constructs such as merged cells. VRDU [6] highlights that hierarchical and long-list fields remain challenging for LLM-based extraction.

We introduce LongListBench, a benchmark designed to stress-test long-list extraction from semi-structured business documents with long incident/line-item lists under systematically injected document phenomena and OCR noise. The benchmark is inspired by recurring patterns observed in real-world claims documents.

1.1 Background and Motivation

In production workflows, a single semi-structured PDF can contain many incidents or line items and associated financial breakdowns. Systems must extract a complete list of records (not merely a few key-value pairs) while handling complex layouts and noisy OCR. We aim to support research on extraction methods that remain reliable as list length grows and as layout artifacts accumulate.

1.2 Research Questions

This work is organized around three practical questions:

- How do common long-list document phenomena (page breaks, duplicates, multi-row cells, multi-column layouts, irrelevant tables, merged cells) affect extraction quality?
- To what extent are end-to-end failures attributable to OCR transcription versus downstream extraction?
- How do strong off-the-shelf LLMs perform under a simple, reproducible zero-shot protocol?

1.3 Contributions

We make the following contributions:

- A reproducible benchmark generation pipeline that produces paired ground truth JSON, rendered PDFs, and OCR transcripts.
- A dataset of 80 documents (40 detailed, 40 table) containing 6,828 incident rows across four difficulty tiers, with an extreme tier reaching 500 incidents per document.

- A taxonomy of seven injected problem types and evaluation scripts for field-level scoring and OCR identifier coverage.
- Baseline results for GPT-5.2, GPT-4o, and Gemini 2.5 under a shared prompt, highlighting remaining gaps in long-list extraction.

1.4 Paper Organization

Section 2 reviews relevant datasets and models. Section 3 describes benchmark construction and the problem taxonomy. Section 4 presents the evaluation protocol. Section 5 reports baseline results. Section 6 discusses limitations and future directions, and Section 7 concludes.

2 Related Work

Research on information extraction (IE) from visually rich documents has produced a broad ecosystem of datasets and models. However, much of the public evaluation landscape emphasizes either short documents (e.g., forms) or key-value extraction, leaving long-list entity extraction underexplored.

2.1 Document IE benchmarks

Early and widely used benchmarks such as FUNSD [3] focus on form understanding in noisy scans. Receipt datasets and challenges such as SROIE [4] emphasize OCR and key fields in narrow document types. These benchmarks are valuable, but typically contain relatively short documents and do not directly stress long lists of repeated entities.

DocILE [5] broadens the scope to business documents and includes line-item recognition, which is closer in spirit to long-list extraction. VRDU [6] further argues that hierarchical and repeated fields (e.g., invoice line items) remain difficult for LLM-based extraction. Our benchmark complements these efforts by focusing on list length, repeated entity boundaries, and a targeted taxonomy of long-list failure modes.

2.2 Document understanding models

Layout-aware pretraining approaches such as LayoutLM [1] jointly model textual content and 2D document structure, yielding strong performance on a range of document understanding tasks. In parallel, OCR-free approaches such as Donut [2] avoid explicit OCR by directly generating structured outputs from document images, mitigating OCR error propagation at the cost of specialized training.

In contrast, our work is model-agnostic: we provide paired PDF, OCR transcript, and ground truth, enabling evaluation of OCR-based pipelines, OCR-free models, and LLM-based extraction. Our primary goal is to support reproducible measurement of long-list extraction robustness under realistic layout artifacts.

3 Benchmark Construction

We construct LongListBench, a synthetic benchmark for long-list entity extraction in semi-structured business documents with long incident/line-item lists, inspired by recurring patterns observed in real-world claims documents. Each benchmark instance consists of (i) structured ground truth incidents (JSON), (ii) a rendered PDF, and (iii) an OCR transcript of the PDF in Markdown.

3.1 Entity schema

Ground truth incidents follow the schema in `benchmarks/models/loss_run.py`, represented as a Pydantic model. The schema includes incident identifiers, policy metadata, narrative text, and nested financial breakdowns (`bi`, `pd`, `lae`, `ded`). While downstream workflows often emphasize fields such as `incident_number`, `company_name`, `date_of_loss`, `status`, `driver_name`, `coverage_type`, and `total_incurred`, our evaluator requires and scores the full schema.

3.2 Document generation

Benchmark instances are generated with seeded randomness for reproducibility (`benchmarks/generate_claims_benchmark.py`). Structured incidents are created by `benchmarks/synthetic/generate_claim_data.py`. We then render the incidents into visually rich documents using `benchmarks/synthetic/generate_html.py` and one of two layouts:

1. **Detailed**: a repeated incident block with narrative text and a small financial table.
2. **Table**: a compact tabular representation formatted as a CSV-like table.

The HTML is rendered into PDF using headless Chromium via Playwright (`benchmarks/synthetic/html_to_pdf.py`).

3.3 Injected problem types

We inject seven recurring document phenomena that complicate long-list extraction (Table 1). These effects are applied at the HTML level prior to PDF rendering.

3.4 Dataset scale

The released dataset (`longlistbench-v1`, version 1.0.1; see `benchmarks/claims/metadata.json`) contains 80 PDFs (40 detailed, 40 table) with 6,828 incident rows. Difficulty tiers are configured as 15 easy instances (10 claims/PDF), 12 medium (25 claims/PDF), 8 hard (50 claims/PDF), and 5 extreme (100 claims/PDF nominal). In the extreme tier, enabling `large_doc` expands each document to 500 incidents. Enabling `duplicates` injects additional duplicate rows (up to 5 per document), causing the number of incident rows to exceed the nominal tier size.

Across the 80 documents, the most common injected issues are multi-row entities (62/80), page breaks (56/80), duplicates (56/80), and multiple tables (40/80).

Table 1: Problem types injected into benchmark documents.

Problem	Description
Page breaks	Split incidents/rows across page boundaries.
Multi-row entities	Insert line breaks inside cells (e.g., descriptions) to stress OCR and parsing.
Exact duplicates	Repeat a subset of incidents verbatim.
Large documents	Expand documents to at least 500 incidents to stress context limits.
Multiple tables	Add irrelevant tables (e.g., company directory) alongside the true claims.
Multi-column layout	Render content in two columns, inducing reading-order ambiguity.
Merged cells	Use row/column spanning and omitted repeated cells in table format.

4 Evaluation

We evaluate systems on the task of extracting a list of incident records from the OCR transcript of each PDF. The benchmark provides both the OCR transcript and a structured JSON ground truth for each document.

4.1 OCR transcription

All PDFs are converted to Markdown using a Gemini vision model (`benchmarks/ocr_claims_pdfs.py`). Each page is rendered to an image and transcribed with a system prompt that emphasizes preserving layout, spacing, and tables. In particular, tables are emitted in a CSV-like form inside Markdown, and the output is concatenated across pages.

4.2 LLM extraction protocol

We provide a lightweight, zero-shot evaluation harness (`benchmarks/evaluate_models.py`) that applies the same extraction prompt to multiple LLM providers and requires the model to return a JSON list of incident objects conforming to the full incident schema. The prompt includes a JSON Schema serialization of the target Pydantic model and is executed at temperature 0. Where supported, we request native structured outputs (e.g., response schemas) to reduce formatting errors.

To ensure schema conformance, model outputs are validated and normalized against a Pydantic schema before scoring. Predictions are stored as `{sample}_{model}_predicted.json`, and aggregate reports are exported as `evaluation_report.json` and `evaluation_report.md`.

4.3 Chunking and merging for long documents

Hard and extreme documents can contain hundreds of incidents, and the OCR transcript may exceed practical context limits. The evaluation harness therefore supports chunked

extraction: the OCR text is split into overlapping chunks using simple incident-number markers, targeting at most eight incidents per chunk. Each chunk is extracted independently, and chunk-level predictions are merged by normalized incident identifier, preferring non-empty fields and combining nested financial breakdown subfields.

4.4 Report regeneration and validation

To support reproducible analysis, the harness can regenerate summary reports offline from saved prediction files and optionally reuse extraction-time values from a previous report. A companion checker script (`benchmarks/check_evaluation_report.py`) recomputes metrics from the saved predictions and the golden data, and flags schema violations or report inconsistencies.

4.5 Field-level matching and metrics

For scoring we use the incident number as the record identifier. Incident numbers are normalized by stripping common prefixes (e.g., #, Incident #). Let G be the list of ground-truth records and P be the list of predicted records. We compute micro precision/recall/F1 over field-value pairs, after canonicalizing each incident under the schema.

Canonicalization strips whitespace from strings, maps empty optional strings to null, sorts claimant lists, and rounds monetary values in nested financial breakdowns to two decimal places. Metrics are computed per document and then averaged across documents for tier-and format-level summaries.

For each incident, we flatten its fields into a multiset of canonicalized triples (`incident_id`, `field_path`, `value`) (including nested financial breakdown fields). We then define:

$$\text{found} = |\mathcal{F}(G) \cap \mathcal{F}(P)|, \quad (1)$$

$$\text{recall} = \frac{\text{found}}{|\mathcal{F}(G)|}, \quad (2)$$

$$\text{precision} = \frac{\text{found}}{|\mathcal{F}(P)|}, \quad (3)$$

$$F1 = \frac{2 \text{precision recall}}{\text{precision} + \text{recall}}, \quad (4)$$

where $\mathcal{F}(\cdot)$ denotes the multiset of flattened field-value pairs across incidents. We additionally report missing and extra incident identifiers and count exact record matches for incidents whose canonicalized objects match exactly.

4.6 OCR identifier coverage baseline

To separate OCR failures from extraction failures, we run a deterministic identifier coverage check (`benchmarks/validate_ocr_vs_golden.py`) which verifies whether incident numbers and reference numbers from the ground truth appear verbatim in the OCR transcript.

Table 2: OCR identifier coverage on the full dataset (80 documents).

Identifier	Mean coverage	Min coverage
Incident number	100.0%	100.0%
Reference number	100.0%	100.0%

Table 3: Zero-shot LLM baseline results across the full benchmark (80 documents) under schema-conformant, field-level scoring (computed from `benchmarks/results_*_all/evaluation_report.json`).

Model	Samples	Avg Recall	Avg Precision	Avg F1
Gemini 2.5	80	80.4%	83.4%	81.9%
GPT-4o	80	78.3%	82.0%	80.0%
GPT-5.2	80	76.8%	79.6%	78.1%

5 Results

We summarize results for (i) OCR fidelity and (ii) baseline extraction performance. OCR coverage and LLM baseline numbers are produced by the released scripts in the repository.

5.1 OCR identifier coverage

Using `benchmarks/validate_ocr_vs_golden.py` we measure how often key identifiers from the ground truth appear verbatim in the OCR transcript. Across the full dataset (80 OCR transcripts), incident numbers and reference numbers exhibit 100% coverage (mean and minimum). These results indicate that, for primary identifiers, our OCR step rarely drops information and that most downstream failures are attributable to extraction rather than transcription.

5.2 Zero-shot LLM extraction baseline

We evaluate three LLMs using the shared prompt and evaluation harness in `benchmarks/evaluate_models`. We report schema-conformant, field-level scoring across the full benchmark (80 documents: 40 detailed, 40 table) using the released per-tier evaluation reports (`benchmarks/results_*_all/evaluation_report.*`). Averaged across all documents, Gemini 2.5 achieves 81.9% average F1 (80.4% recall, 83.4% precision), GPT-4o achieves 80.0% average F1 (78.3% recall, 82.0% precision), and GPT-5.2 achieves 78.1% average field-level F1 (76.8% recall, 79.6% precision) (Table 3). Across all models, the detailed format is substantially easier than the table format (Table 4), and performance varies meaningfully across difficulty tiers (Table 5).

Qualitatively, errors often manifest as local field-level deviations (e.g., missing optional strings, numeric drift in financial breakdowns, or small identifier formatting mistakes) spread across an otherwise correct long list.

These findings suggest that recovering identifiers is largely deterministic under our OCR pipeline, while the main open challenge for long-list extraction is robustly segmenting and

Table 4: Baseline F1 by document format aggregated across all tiers (`benchmarks/results_*_all/evaluation_report.json`).

Model	Detailed F1	Table F1
Gemini 2.5	89.8%	73.9%
GPT-4o	89.3%	70.8%
GPT-5.2	83.5%	72.8%

Table 5: Baseline F1 by difficulty tier (average across documents within each tier; `benchmarks/results_*_all/evaluation_report.json`).

Tier	Samples	Gemini 2.5 F1	GPT-4o F1	GPT-5.2 F1
Easy	30	85.1%	82.2%	80.2%
Medium	24	80.5%	79.7%	76.7%
Hard	16	78.1%	76.6%	76.0%
Extreme	10	81.6%	80.0%	78.9%

populating full per-incident records under layout disruptions (page breaks, multi-column order, irrelevant tables, merged cells) and scale (hundreds of incidents).

6 Limitations and Future Directions

6.1 Limitations

LongListBench is designed to be a practical benchmark for long-list extraction in semi-structured business documents, but it has several limitations.

- **Synthetic generation:** while the schema and failure modes are motivated by production documents, the records and layouts are programmatically generated. This may under-represent rare formatting conventions and highly idiosyncratic carrier templates.
- **OCR stack:** OCR transcripts are produced by a single vision-language model with a fixed prompt. Classical OCR systems and alternative prompts may yield different error distributions.
- **Domain scope:** the current instances are inspired by recurring patterns observed in real-world claims documents. Generalization to other long-list domains (invoices, purchase orders, medical billing, financial statements) should be validated.
- **Metric granularity:** while we report schema-conformant, field-level scoring, the metric does not capture semantic equivalence or provide exact-match analyses, and it remains challenging to evaluate duplicates as first-class entities.

6.2 Future directions

We see multiple immediate extensions that would strengthen LongListBench and increase its utility.

- **Per-field analysis:** report more detailed per-field accuracy for the full incident schema, and explicitly evaluate duplicate detection and normalization.
- **Exact-match analysis:** evaluate the ability of models to produce exact matches for extracted fields.
- **Broader OCR conditions:** add scans with blur/noise, different DPI settings, and non-LLM OCR baselines.
- **Broader document families:** add additional templates and long-list document types beyond the claims-inspired formats in the current release.
- **Scalable extraction protocols:** benchmark chunking, retrieval-augmented extraction, and layout-aware reconstruction strategies for the extreme tier (500 incidents).

7 Conclusion

LongListBench targets a persistent gap in document understanding evaluation: extracting long lists of repeated entities from semi-structured business documents under realistic layout and OCR noise. We presented a benchmark construction pipeline that produces paired (PDF, OCR, JSON) artifacts and systematically injects common long-list failure modes.

7.1 Summary

Our main contributions are:

- A reproducible benchmark generation pipeline for semi-structured documents with long incident lists spanning two formats and four difficulty tiers.
- A taxonomy of seven problem types that frequently break long-list extraction systems, including duplicates, page breaks, multi-row entities, multi-column layout, and merged cells.
- An evaluation harness and baseline results that quantify the gap between near-perfect OCR identifier retention and imperfect end-to-end extraction.

7.2 Practical takeaways

We intend LongListBench to be useful as a measurement tool for both research and engineering workflows. Two practical takeaways are worth emphasizing. First, identifier retention in OCR is near-perfect (Table 2), so most end-to-end failures should be attributed to downstream parsing, segmentation, and field population rather than transcription. Second,

even with schema-conformant structured outputs and a shared prompt, field-level extraction across the full benchmark remains materially below perfect (Table 3), with a large gap between detailed and table formats (Table 4) and meaningful variation across difficulty tiers (Table 5), indicating substantial headroom for methods that explicitly model reading order, table structure, and long-range consistency.

7.3 Recommended reporting

For comparability across papers and systems, we recommend that LongListBench results report (i) OCR identifier coverage, (ii) schema-conformant field-level precision/recall/F1 under the released evaluator, and (iii) the extraction protocol used for long documents (e.g., full-context vs chunking, chunk sizes, and merge strategy). The extreme tier, in particular, is intended to stress scaling behavior: methods that succeed on short lists may fail due to context limits, brittle segmentation, or accumulated small errors across hundreds of records.

7.4 Future Work

We view the benchmark as a foundation for studying scalable, layout-robust extraction. Immediate next steps include improved handling of duplicates and merged cells, and evaluation of methods that can reliably extract hundreds of incidents in a single document (Section 6).

Acknowledgments

We thank Ben Perryman and Colin for helpful feedback on early drafts and for reviewing the manuscript.

References

- [1] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, *Layoutlm: Pre-training of text and layout for document image understanding*, 2020. DOI: [10.1145/3394486.3403172](https://doi.org/10.1145/3394486.3403172). arXiv: [1912.13318 \[cs.CL\]](https://arxiv.org/abs/1912.13318). [Online]. Available: <https://arxiv.org/abs/1912.13318>.
- [2] G. Kim et al., *Ocr-free document understanding transformer*, 2022. arXiv: [2111.15664 \[cs.LG\]](https://arxiv.org/abs/2111.15664). [Online]. Available: <https://arxiv.org/abs/2111.15664>.
- [3] G. Jaume, H. K. Ekenel, and J.-P. Thiran, *Funsd: A dataset for form understanding in noisy scanned documents*, 2019. arXiv: [1905.13538 \[cs.IR\]](https://arxiv.org/abs/1905.13538). [Online]. Available: <https://arxiv.org/abs/1905.13538>.
- [4] Z. Huang et al., *Icdar2019 competition on scanned receipt ocr and information extraction*, 2021. DOI: [10.1109/ICDAR.2019.00244](https://doi.org/10.1109/ICDAR.2019.00244). arXiv: [2103.10213 \[cs.AI\]](https://arxiv.org/abs/2103.10213). [Online]. Available: <https://arxiv.org/abs/2103.10213>.
- [5] Š. Šimsa et al., *Docile benchmark for document information localization and extraction*, 2023. arXiv: [2302.05658 \[cs.CL\]](https://arxiv.org/abs/2302.05658). [Online]. Available: <https://arxiv.org/abs/2302.05658>.

- [6] Z. Wang, Y. Zhou, W. Wei, C.-Y. Lee, and S. Tata, “Vrdu: A benchmark for visually-rich document understanding,” in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, arXiv:2211.15421, 2023. doi: [10.1145/3580305.3599929](https://doi.org/10.1145/3580305.3599929). [Online]. Available: <https://arxiv.org/abs/2211.15421>.