Problem Solved? Information Extraction Design Space for Layout-Rich Documents using LLMs

Gaye Colakoglu[1,2] **Gürkan Solmaz**[2] **Jonathan Fürst**[1]
[1]Zurich University of Applied Sciences, Switzerland
[2]NEC Laboratories Europe, Heidelberg, Germany
colgay01@students.zhaw.ch, guerkan.solmaz@neclab.eu, jonathan.fuerst@zhaw.ch

Abstract

This paper defines and explores the design space for information extraction (IE) from layout-rich documents using large language models (LLMs). The three core challenges of layout-aware IE with LLMs are 1) data structuring, 2) model engagement, and 3) output refinement. Our study investigates the subproblems and methods within these core challenges, such as input representation, chunking, prompting, selection of LLMs, and multimodal models. It examines the effect of different design choices through *LayIE-LLM*, a new, open-source, layout-aware IE test suite, benchmarking against traditional, fine-tuned IE models. The results on two IE datasets show that LLMs require adjustment of the IE pipeline to achieve competitive performance: the optimized configuration found with LayIE-LLM achieves 13.3–37.5 F1 points more than a general-practice baseline configuration using the same LLM. To find a well-working configuration, we develop a one-factor-at-a-time (OFAT) method that achieves near-optimal results. Our method is only 0.8–1.8 points lower than the best full factorial exploration with a fraction (∼2.8%) of the required computation. Overall, we demonstrate that, if well-configured, general-purpose LLMs match the performance of specialized models, providing a cost-effective, finetuning-free alternative. Our test-suite is available at https://github.com/gayecolakoglu/LayIE-LLM.

1 Introduction

Information extraction (IE) extracts structured data from unstructured documents, including layout-rich documents (LRDs) as reports and presentations that mix visual and textual elements (Park et al., 2019; Wang et al., 2023b; Zmigrod et al., 2024b). LRDs challenge traditional natural language processing (NLP) techniques, designed for plain texts (Cui et al., 2021; Tang et al., 2023).

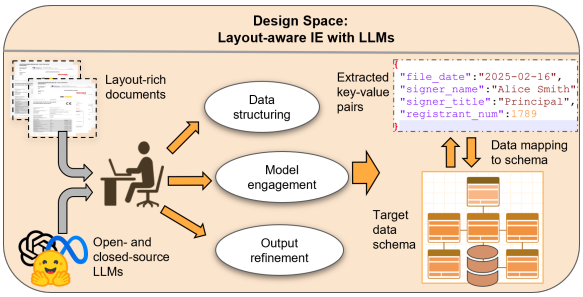

Figure 1: Design space for IE from layout-rich documents using LLMs. The goal is to extract information relevant to the target data schema with correct mapping.

Recent layout-aware models at the intersection of NLP and Computer Vision (CV) address this gap by including visual and structural features to improve IE from LRDs (e.g., LayoutLMv1-v3 (Xu et al., 2020b,a; Huang et al., 2022), ERNIE-Layout (Peng et al., 2022), GraphDoc (Zhang et al., 2022), DocFormer (Appalaraju et al., 2021)). While early layout-aware models required substantial dataset-specific fine-tuning, recent generative document understanding models (Powalski et al., 2021) have demonstrated strong zero- and few-shot capabilities on unseen datasets and tasks.

While LLMs exhibit significant potential, there exist open questions regarding their adoption for IE from LRDs. In what form should the document content be provided to the LLM? Which methods are most effective for in-context learning (ICL) and instruction tuning? Do large multimodal LLMs as GPT-4o (OpenAI, 2024) and Qwen 2.5-vision (Bai et al., 2025) offer advantages over traditional Optical Character Recognition (OCR) and fine-tuned layout-aware models, and how does their performance vary across LLMs of different scales and sources? How coherent is the LLM output, and what post-processing is needed?

Studies have individually addressed some of the aspects (Wang et al., 2023a; Luo et al., 2024a; Blau et al., 2024). However, there is a lack of a unified

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 17908–17927
November 4-9, 2025

framework that comprehensively explores and compares these components. This paper fills this gap by systematically investigating the design space for LLM-based IE from LRDs. We focus on evaluating a range of preprocessing, chunking, prompting, and post-processing techniques, alongside various ICL strategies, within a unified, reproducible framework. Additionally, we conduct an extensive comparison of these components, examining multimodal LLMs, traditional models, open- and closed-source LLMs, and fine-tuned layout-aware models. *To the best of our knowledge, no existing research has comprehensively compared the combined impact of these pipeline parameters across such diverse model categories.*

Our results reveal multiple insights. First, current general LLMs can compete with traditional fine-tuned models such as LayoutLMv3 and ERNIE-Layout, without expensive labeling. Second, rather than fine-tuning on data, LLMs require adjustment of the IE pipeline to achieve competitive performance–the performance gap between a general-practice baseline and the configuration tuned using our lightweight OFAT method on two datasets is 13.3 F1 points for VRDU (Wang et al., 2023c), and 37.5 for FUNSD (Jaume et al., 2019), only 1.8 points and 0.8 points lower than the best possible Brute-Force configuration respectively. Third, while purely text-based LLMs achieve competitive performance with our method, multimodal LLMs, those that directly integrate textual and visual features, still outperform them. However, this advantage comes with increased token usage, higher API costs, and diminished transparency and control over individual steps. In summary, our contributions are as follows.

- We introduce the **Design Space of IE from LRDs** using LLMs, addressing three core challenges: 1) Data structuring, 2) Model engagement, and 3) Output refinement (Sec. 2).
- We develop LayIE-LLM, a **layout-aware IE test suite** to analyze the impact of: OCR, text-based inputs, chunk size, few-shot and CoT prompting, LLM models, decoding strategies, schema mapping, data cleaning, and evaluation techniques using exact, substring, and fuzzy matching (Sec. 3).
- We conduct evaluation of **GPT-4o**, **GPT-3.5**, and **LLaMA3**, as well as the vision-enabled **GPT-4o and Qwen2.5-vision**, and compare them with traditional, fine-tuned layout-aware models such as **LayoutLMv3** and **ERNIE-Layout**, highlighting trade-offs between performance and resource efficiency (Sec. 4).
- We open-source LayIE-LLM and all our experimental results, enabling reproducibility and further experimentation by the community: https://github.com/gayecolakoglu/LayIE-LLM

2 Design Space of IE from LRDs with LLMs

Task Definition. IE from LRDs involves identifying and extracting information from documents where textual content is intertwined with complex visual layouts and mapping them into structured information instances such that

$$\text{IE} : (D, S) \rightarrow E \tag{1}$$

- **D** represents the set of LRDs, each with content and layout information.
- **S** is the target schema that defines the set of slots to be filled. Each slot is defined by an attribute (key) a_i and its corresponding data type (domain) T_i, such that $S = \{(a_1, T_1), (a_2, T_2), ..., (a_k, T_k)\}$.
- Finally, **E** represents the set of extracted information instances, where each instance is a set of slot-value pairs derived from a document in **D**, leveraging both content and layout to determine the correct values for the slots in **S**. Each value in an instance must conform to the data type T, specified in the schema for that attribute.

2.1 Using LLMs for IE

LLM-based IE systems have to tackle three challenge areas: *Data Structuring*, *Model Engagement*, and *Output Refinement*.

Data Structuring. For multimodal LLMs, LRDs can be directly given as input. On the other hand, for purely text-based LLMs, the input documents must be transformed into textual representations. This involves converting documents into machine-readable formats using OCR systems to extract features such as text, bounding boxes, and visual elements (Mieskes and Schmunk, 2019; Smith, 2007). Alternatively, a formatting language such as Markdown can be employed to represent the document's layout, allowing the LLM to understand the structural context of the text better. The impact of OCR quality on IE performance has been documented (Bhadauria et al., 2024), and structured formats tend to yield better results (Bai et al., 2024). To process larger

documents efficiently, they are often divided into smaller, manageable chunks based on page boundaries, sections, or semantic units (Liu et al., 2024). *Markdown as an input format compared to raw OCR outputs remains underexplored, representing a potential research gap in IE system development.*

Model Engagement. Once preprocessed, the document is fed to an LLM for IE. Ensuring alignment between the extracted text and layout information is crucial for accurate representation (Xu et al., 2020a; Appalaraju et al., 2021). Prompt-driven extraction leverages general-purpose models, by using tailored prompts to guide the extraction process (Brown et al., 2020; Radford et al., 2021; Zhou et al., 2022). As such, models must be explicitly instructed to extract information, typically with an accompanying schema. This step can involve more advanced IT and ICL techniques (few-shot, CoT). *The influence of prompting techniques in interaction with various stages of the IE pipeline to enhance performance and robustness remains a research gap that requires further investigation.*

Output Refinement. After inference, the extracted information undergoes post-processing to ensure accuracy and conformity with Schema S. This step involves refining and validating the outputs generated by the LLM through tasks such as mapping extracted entities E to their original document positions, merging overlapping or fragmented predictions, and resolving ambiguities in the results (Xu et al., 2020b). Post-processing for entity extraction has been explored in prior studies (Wang et al., 2022b; Tamayo et al., 2022), and rule-based entity alignment has demonstrated notable improvements in accuracy (Luo et al., 2024b). *However, to our knowledge, there has been no analysis of post-processing techniques specifically tailored to LRDs in conjunction with LLMs.*

3 LayIE-LLM: Test-Suite for IE from LRDs with LLMs

We implement LayIE-LLM , a comprehensive test suite to evaluate IE tasks from LRDs. The pipeline, depicted in Figure 2, systematically transforms raw inputs into structured outputs through multiple stages, enabling a thorough assessment of the effectiveness of various design decisions.

3.1 Data Structuring

PDF documents include both textual content and layout features. The process involves two conversions: 1) Extracting textual content and layout information using OCR data, and 2) creating a markdown representation.

Chunking. We employ three chunk sizes: (1) max: 4096 tokens, (2) medium: 2048 tokens, and (3) small: 1024 tokens. Documents are segmented into N chunks based on their length and the selected chunk size. Each chunk is constructed by sequentially accumulating whole words up to the token limit, thereby preserving word boundaries and ensuring a non-overlapping structure. Layout information is retained by associating each text segment with normalized and quantized spatial coordinates, which preserve the structural context of the original document.

3.2 Model Engagement

Model engagement involves constructing input to the LLM using three primary components: (1) a task instruction that defines the IE task, (2) the target schema S, and (3) the document chunk. We adhere to best practices from NLP for prompt structure and IE task instruction. The schema S is implemented as a dictionary of key-value pairs, where values specify the format of the corresponding attribute using regex expressions in Listing 1.

```
"file_date": r"\d{4}-\d{2}-\d{2}",
"foreign_princ_name": r"[\w\s.,'&-]+",
"registrant_name": r"[\w\s.,'&-]+",
"registration_num": r"\d+",
"signer_name": r"[\w\s'.-]+",
"signer_title": r"[\w\s.,'&-]+",
```

Listing 1: Schema Definition

We also implement two ICL strategies: (1) Few-Shot and (2) Chain-of-Thought (CoT) (see Appendix A.2 for more details). For N prompts, the LLM performs *N completions*, with one completion per prompt. The outputs are collected and stored as raw predictions, ready for Output Refinement.

3.3 Output Refinement

We refine the raw outputs from the LLM to ensure alignment with the target schema, addressing challenges related to prediction variability, schema definition differences, and data formatting inconsistencies (e.g., varying date formats). We implement three techniques inspired by related work in data integration (Dong and Srivastava, 2013): *Decoding*, *Schema Mapping*, and *Data Cleaning*. This results in three sets of predictions: initial predictions, mapped predictions, and cleaned predictions.

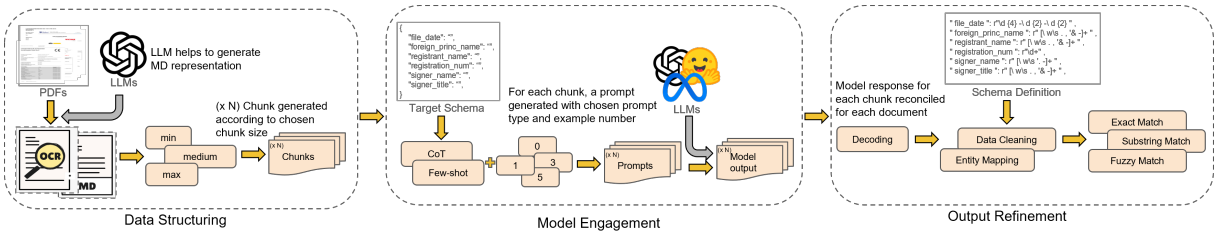

Figure 2: LayIE-LLM test suite for extracting information from LRDs using LLMs in six stages. The process begins with OCR-based text extraction and Markdown conversion with LLM assistance, followed by chunking to manage token limits, experimenting with different chunk sizes. Each chunk is processed into a prompt using Few-shot and CoT strategies with varying example counts. Prompts include a document example with key-value pairs, plus a new document and task. LLMs generate structured JSON outputs, which are decoded and reconciled. Post-processing includes data cleaning and entity mapping, followed by evaluation using two methods.

Decoding. The decoding step parses each LLM completion as a JSON object, discarding any that fail to parse. The process then consolidates predictions for each document by reconciling outputs generated across individual pages and chunks. With N completions for N prompts, corresponding to N chunks, the model generates multiple predictions for a single document. Reconciliation ensures a unified document-level output by deduplicating nested predictions and aggregating unique values. If multiple unique values exist for a single entity, they are retained to preserve variability. The outcome of this step is referred to as *initial predictions*.

Schema Mapping. LLMs are expected to return only keys $\{a_1, a_2, ..., a_k\}$ specified in the target schema S. However, they may occasionally fail to return the keys as expected. E.g., 'file date" is returned instead of "file_date". Such LLM "overcorrection" can hinder strict schema conformance. As a countermeasure, we implement a post-processing step that maps the predicted keys to align with the target schema. Our mapping step integrates multiple weak-supervision signals, such as *exact matching, partial matching,* and *synonym-based logic*, inspired by the recent techniques for ontology alignment (Fürst et al., 2023). The outcome of this step is the *mapped predictions*, where entity keys $\{a_1, a_2, ..., a_k\}$ are standardized and fully aligned with the schema S.

Data Cleaning. A common issue concerns the format of the values T_k of predicted key-value pairs $\{(a_1, T_1), (a_2, T_2), ..., (a_k, T_k)\}$. We must standardize formats, such as dates and names, to align with the target schema S. One source of error is LLM hallucinations (Ji et al., 2023; Huang et al., 2023; Xu et al., 2024b), while another problem is that information is often not aligned to a common format inside the source data. For instance, two documents might use two different formats for dates ("April 1992" vs "1992-04-01"). Additional issues include capitalization, redundant whitespace, or special characters. We utilize the regex-defined data types in our schema to automatically apply data cleaning functions. The outcome of this step is the *cleaned predictions*, representing the final fully normalized outputs.

Evaluation Techniques. Evaluating IE for LRDs requires comparing the extracted data against an annotated test dataset. We implement three metrics for this evaluation: *exact match, substring match*, and *fuzzy match*.

- **Exact Match** searches for perfect alignment between predicted and ground truth values. A match is valid only when the values are identical. This strict approach is ideal for extracting specific, unambiguous entities like dates or numerical identifiers.
- **Substring Match** checks whether ground truth values are fully contained within the predicted values as complete substrings, without being split or partially matched. It ensures all ground truth values appear in their entirety within predictions, making it effective for tasks such as extracting full names or addresses, where additional contextual details (e.g., titles like Mr. and Mrs.) may be included in the predictions without making the extraction incorrect.
- **Fuzzy Match** uses similarity metrics (fuzz.ratio) for approximate matches. A match is valid if the highest similarity ratio exceeds a predefined threshold (default: 0.8), creating binary decisions unlike soft similarity metrics (Biten et al., 2019). This method is well-suited for scenarios with minor variations caused by OCR errors or for-

matting discrepancies.

4 Experimental Evaluation

4.1 Experimental Design

Methodology. The goal of our experimental setup is to analyze how different parameters in the pipeline (Figure 2) influence the performance of IE from LRDs using LLMs. To examine the design dimensions, we begin with a baseline configuration and systematically vary one factor at a time, following a one-factor-at-a-time (OFAT) methodology. This approach allows us to isolate and understand the impact of each parameter change on the IE performance.

Our intuition is that by independently aggregating the insights gained from each design dimension, one can develop a deeper understanding of the design space and potentially identify an effective overall configuration for IE, without the need for an exhaustive factorial exploration. To validate this approach, we compare the OFAT method's results with those obtained via a brute-force strategy involving 432 experiments ($2\times3\times2\times4\times3\times3 = 432$, see Table 1).

Dataset and LLMs. We use the Visually Rich Document Understanding (VRDU) dataset (Wang et al., 2023c), which includes two benchmarks with high-quality OCR for assessing data efficiency, due to its inclusion of three visually diverse template types – Amendment, Short-Form, and Dissemination – representing varied and complex layouts, as well as generalization tasks: *Single Template Learning (STL), Unseen Template Learning (UTL)*, and *Mixed Template Learning (MTL)*. For further details on how this dataset is tailored for our experiments and diverse models, please refer to A.3.

Additionally, to further prove the generalizability of our approach, we also conducted the same set of experiments using the FUNSD dataset (Jaume et al., 2019), which directly provides key-value pairs in a question-answering format without the complexity of OCR processing. Due to page limitations and for the sake of clarity, the experimental results presented in this section are based on the VRDU dataset. The corresponding experimental results for the FUNSD dataset are provided in A.5.

We evaluate GPT-3.5, GPT-4o, and LLaMA3-70B for text-only structured data extraction from LRDs. Furthermore, we compare their results with those of GPT-4, LayoutLMv3, and ERNIE-Layout to assess the performance gap between multimodal LLMs and domain-specific fine-tuned models.

Table 1: Overall configuration parameters. Baseline configuration is highlighted with light blue.

Parameter	Values
Input Type	OCR, Markdown
Chunk Size Category	Small, Medium, Max
Prompt Type	Few-Shot, CoT
Example Number	0, 1, 3, 5
Post-processing Strategy	Initial, Mapped, Cleaned
Evaluation Technique	Exact, Substring, Fuzzy

The Baseline Configuration. The baseline configuration is outlined in Table 1, where the configuration is selected based on best practices, such as in (Perot et al., 2024) for the following reasons: (1) **OCR** reflects real-world scenarios for digitized LRDs. (2) **Medium chunk size** balances efficiency and context preservation, addressing token limits in LLMs. (3) **Few-shot prompting** combines pretrained knowledge with minimal task-specific guidance. (4) Using **zero examples** provides a clear benchmark for assessing the model's raw performance. (5) **Initial predictions** are retained to evaluate models' raw output without modifications, ensuring a direct assessment of their capabilities. (6) Finally, **exact match** provides a stringent measure of correctness, offering a reliable baseline for comparison across configurations.

4.2 The Input Dimension

We substitute OCR input with Markdown and evaluate the resulting performance in both STL and UTL scenarios. The performance differences between OCR and Markdown inputs vary by model and context, showing no consistent trend favoring either input type, as shown in Table 2.

Table 2: Performance results for different LLMs across **STL** and **UTL** levels with different input types. Baseline configuration in light blue.

Models	Level	Exact Match (F1)	
		OCR	Markdown
GPT-3.5	STL	0.650	0.647 (-0.003)
	UTL	0.645	0.657 (+0.012)
GPT-4o	STL	0.670	0.633 (-0.037)
	UTL	0.659	0.633 (-0.026)
LLaMA3	STL	0.640	0.657 (+0.017)
	UTL	0.640	0.662 (+0.022)
Avg (±stdev.)		0.650 (±0.011)	0.648 (±0.012)

OCR input serves as a stable baseline for IE tasks, delivering consistent performance across models. GPT-4o has noticeable performance drops with Markdown input, indicating its reliance on OCR for optimal results. In contrast, Markdown marginally improves performance for LLaMA3-70B at both STL and UTL scenarios, suggesting its potential benefits from the additional structure or semantic cues. GPT-3.5 demonstrates robustness to changes in input type, with only slight fluctuations in performance. On average, OCR marginally outperforms Markdown (0.650 vs. 0.648), but the differences are minor, with standard deviations indicating similar stability (see Appendix A.1 for detailed insights).

4.3 The Chunk Dimension

Table 3: Performance results for different LLMs across **STL** and **UTL** levels with different chunk size categories. Baseline configuration in light blue.

Models	Level	Exact Match (F1)		
		Small($\leq$ 1024)	**Medium**($\leq$ 2048)	**Max**($\leq$ 4096)
GPT-3.5	STL	0.562(-0.088)	0.650	0.645(-0.005)
	UTL	0.561(-0.084)	0.645	0.644(-0.001)
GPT-4o	STL	0.602(-0.068)	0.670	0.674(+0.004)
	UTL	0.600(-0.059)	0.659	0.657(-0.002)
LLaMA3	STL	0.615(-0.025)	0.640	0.647(+0.007)
	UTL	0.608(-0.032)	0.640	0.644(+0.004)
Avg(±stdev.)		0.591(±0.023)	0.650(±0.011)	0.651(±0.011)

To evaluate the impact of chunk size, we vary it while keeping all other parameters constant. Table 3 demonstrates how chunk size affects performance across STL and UTL levels.

Medium and max chunk sizes provide the most consistent and stable results across models, with an average F1 score of 0.650 (±0.011) and 0.651 (±0.011), respectively. Due to insufficient context, small chunk sizes result in significant performance drops, particularly for GPT-3.5 and GPT-4o. *These findings suggest that max chunk size is optimal, but medium can be a good option for LLMs with limited context lengths.*

4.4 The Prompt Dimension

Table 4 shows the impact of prompt type and number of examples on model performance for STL and UTL. Surprisingly, in-context demonstrations do not enhance performance for few-shot or CoT experiments (see Appendix A.1 for detailed insights). For both experiments, the setting with zero examples achieves the highest average performance: few-shot 0.650 (±0.011) and CoT 0.649 (±0.008). Performance consistently declines as the number of examples increase, likely due to noise that impairs generalization. *Overall, there is no significant difference between few-shot and CoT.*

Table 4: Different LLMs across **STL** and **UTL** levels with different prompt types and example numbers. Baseline Configuration is highlighted in light blue.

Models	Level	Exact Match (F1)			
		0	**1**	**3**	**5**
		few-shot			
GPT-3.5	STL	0.650	0.586(-0.064)	0.593(-0.057)	0.548(-0.102)
	UTL	0.645	0.566(-0.079)	0.564(-0.081)	0.541(-0.104)
GPT-4o	STL	0.670	0.608(-0.062)	0.602(-0.068)	0.595(-0.075)
	UTL	0.659	0.597(-0.062)	0.607(-0.052)	0.601(-0.058)
LLaMA3	STL	0.640	0.599(-0.041)	0.606(-0.034)	0.603(-0.037)
	UTL	0.640	0.582(-0.058)	0.601(-0.039)	0.597(-0.043)
Avg(±stdev.)		0.650(±0.011)	0.589(±0.014)	0.595(±0.016)	0.580(±0.028)
		CoT			
GPT-3.5	STL	0.653(+0.003)	0.602(-0.048)	0.544(-0.106)	0.533(-0.117)
	UTL	0.650(+0.005)	0.575(-0.007)	0.548(-0.097)	0.516(-0.129)
GPT-4o	STL	0.655(-0.015)	0.615(-0.055)	0.612(-0.058)	0.605(-0.065)
	UTL	0.659(0)	0.614(-0.045)	0.611(-0.048)	0.607(-0.052)
LLaMA3	STL	0.635(-0.005)	0.603(-0.037)	0.613(-0.027)	0.610(-0.003)
	UTL	0.644(+0.004)	0.586(-0.054)	0.601(-0.039)	0.598(-0.042)
Avg(±stdev.)		0.649(±0.008)	0.599(±0.015)	0.588(±0.032)	0.578(±0.042)

4.5 Output Refinement

We examine two output refinement strategies, Schema Mapping and Data Cleaning (see Sec. 3.3), to evaluate their impact shown in Table 5.

Schema Mapping involves mapping the predicted schema keys to the target schema keys. Our results show no change in F1 scores compared to the initial predictions. This suggests that the models already effectively return the correct attributes, making the mapping step unnecessary.

Data Cleaning uses the defined data types to perform automatic value cleaning, consistently achieving the highest F1 scores across all models. *This underscores the need for post-processing steps for IE with LLMs to align the extracted data with the target format to handle LLM hallucinations and inconsistent source data formats (see Sec. 3.3).*

4.6 Evaluation Techniques

We explore three evaluation techniques to assess their impact on model performance (see Sec. 3.3). On average, Fuzzy Match achieved the highest F1 score (0.733), outperforming Substring Match (0.676) and Exact Match, as shown in Table 6.

Table 5: Different LLMs across **STL** and **UTL** levels with different post-processing strategies. Baseline configuration is highlighted in light blue.

Models	Level	Exact Match (F1)		
		Initial Pred.	Mapped Pred.	Cleaned Pred.
GPT-3.5	STL	0.650	0.650(0)	0.737(+0.087)
	UTL	0.645	0.645(0)	0.733(+0.088)
GPT-4o	STL	0.670	0.670(0)	0.749(+0.079)
	UTL	0.659	0.659(0)	0.741(+0.082)
LLaMA3	STL	0.640	0.640(0)	0.724(+0.084)
	UTL	0.640	0.640(0)	0.725(+0.085)
Avg(±stdev.)		0.650(±0.011)	0.650(±0.011)	0.734(±0.009)

To validate the reliability of these metrics, we manually analyzed cases where Exact Match failed but Substring or Fuzzy Match succeeded. This review confirmed that most soft-matched predictions were semantically correct, with errors largely due to OCR noise, formatting differences, or annotation inconsistencies. Notably, fuzzy match (with a strict 0.8 threshold) captured valid predictions that exact and substring match missed, such as truncated organization names. Our detailed error analysis in Appendix A.4 shows that fuzzy and substring matching provide near-perfect precision when manually checked for semantic equivalence, with precision scores of 0.98 and 1.00, respectively. *This shows Fuzzy Match's ability to balance flexibility and precision.*

Table 6: Different LLMs across **STL** and **UTL** levels with different evaluation techniques. Baseline configuration in light blue.

Models	Level	Exact Match (F1)	Substring Match (F1)	Fuzzy Match (F1)
GPT-3.5	STL	0.650	0.683(+0.033)	0.730(+0.080)
	UTL	0.645	0.682(+0.037)	0.726(+0.081)
GPT-4o	STL	0.670	0.690(+0.020)	0.750(+0.080)
	UTL	0.659	0.678(+0.019)	0.744(+0.085)
LLaMA3	STL	0.640	0.661(+0.021)	0.727(+0.087)
	UTL	0.640	0.662(+0.022)	0.723(+0.083)
Avg(±stdev.)		0.650(±0.011)	0.676(±0.011)	0.733(±0.010)

4.7 Putting it All Together

We investigated the influence of various parameters on model performance along the IE extraction pipeline, analyzing one factor at a time. Drawing from the underlying 12 experiments, we identified the optimal parameter for each step and each model based on the experimental outcomes (Table 7). To validate this approach, we conducted an exhaustive full factorial exploration with 432 configurations to identify the overall best-performing (Table 8) and worst-performing parameter combinations per LLM. For the comparative analysis in Figure 3, we use a single global OFAT configuration that achieves the highest average performance across all models (selection criteria in Appendix A.1). The performance of these different configurations, including the worst, is depicted in Figure 3. We gain several insights:

- *OFAT approximates well the best-performing Brute-Force configuration with a fraction ($\sim 2.8\%$) of the required computation.* We see in Table 7 and Table 8 that they match except for 8 parameter choices (`chunk size`, `prompt` and `example No.` parameters). Likewise, their F1 scores are close to each other (Figure 3), with OFAT achieving 0.783 and Brute-Force achieving 0.801 overall.
- *Adapting the IE pipeline to the LLM is crucial for competitive performance.* Overall, the OFAT configuration improves from baseline F1 of 0.650 to 0.783, a 20% improvement. For the best-performing Brute-Force, the improvement is 23%. In comparison, the worst configuration achieves an average score of only 0.5, which is approximately 64.1% of the best configuration's performance.
- *Common patterns across LLMs.* First, output refinement and evaluation techniques boost performance for all LLMs. Second, there is a trend towards larger context sizes. Lastly, larger models (GPT-4o, LLaMa3) benefit more from examples, while the CoT pattern generally aids IE.

FUNSD Dataset. We provide the equivalent OFAT analysis for the FUNSD dataset in Appendix A.5. While we observed different patterns with respect to the IE pipeline configuration, our experiments on FUNSD also show a clear improvement through output refinement techniques and the use of tolerant evaluation metrics. Most importantly, they show that our OFAT method nearly reaches the best-performing Brute-Force performance (only 0.8 F1 points lower), and improves 90% from the baseline IE results. *This indicates that our OFAT method for IE pipeline configuration is generalizable and can help practitioners and researchers to find the best configuration for their specific dataset.*

Finally, we compare our text-based approach to IE to (1) GPT-4o-vision and Qwen2.5-vision, multimodal LLMs, (2) LayoutLMv3 and ERNIE-

Table 7: OFAT configurations on a per-model basis and corresponding performance results.

Parameter	GPT-3.5	GPT-4o	LLaMA3
Input Type	Markdown	OCR	Markdown
Chunk Size	Medium	Max	Max
Prompt	CoT	Few-Shot	Few-Shot
Example No.	0	0	0
Output Refin.	Cleaned	Cleaned	Cleaned
Evaluation	Fuzzy	Fuzzy	Fuzzy

Table 8: Best-performing Brute-Force configurations on a per-model basis and corresponding performance results.

Parameter	GPT-3.5	GPT-4o	LLaMA3
Input Type	Markdown	OCR	Markdown
Chunk Size	Max	Medium	Small
Prompt	Few-shot	CoT	CoT
Example No.	0	5	5
Output Refin.	Cleaned	Cleaned	Cleaned
Evaluation	Fuzzy	Fuzzy	Fuzzy

Layout, layout-aware models that we fine-tune to our dataset (Table 9). Despite fine-tuning, LayoutLMv3 and ERNIE-Layout perform worse than our evaluated LLMs, even on documents that contain the same structure as the fine-tuning data (STL). We also observe that LLMs remain stable across template types. The best-performing models are GPT-4o-vision followed by Qwen2.5-vision, both multimodal LLMs, where we directly provide an image of the PDF for IE. These models benefits from less context loss due to chunking and their ability to use textual and visual features jointly. However, when considering token usage and cost, GPT-4o-vision requires $\sim 2\times$ the tokens compared to text-only approaches and $> 10\times$ the cost under the current OpenAI pricing scheme (Nov. 2024), while Qwen2.5-vision uses $\sim 1.4\times$ the tokens with no API costs when run locally. Therefore, text-only approaches constitute a good trade-off between performance and cost for practical applications, with Qwen2.5-vision offering a viable middle ground.

5 Related Work

IE using LLMs. Transformer-based models (Devlin et al., 2019; Brown et al., 2020; Liu et al., 2019b) have advanced NLP with self-attention and large-scale pretraining but struggle with layout-rich documents (LRDs). To address this, layout-aware and multimodal models have emerged (Xu et al., 2020b, 2021, 2022), including generative approaches (Powalski et al., 2021). Additionally,

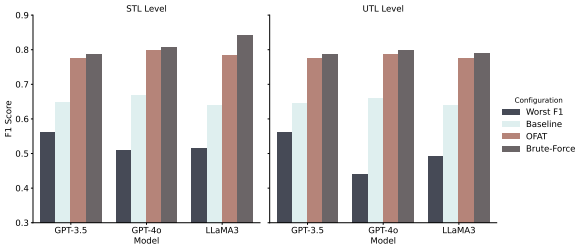

Figure 3: F1 scores of different LLMs across three configurations (Baseline, OFAT, and Brute-Force). Each bar represents the mean F1 score of a model for the corresponding configuration. The OFAT configuration uses a single global parameter set optimized across all models.

Table 9: Cost (per experiment) and performance of LLMs across **STL** and **UTL** levels for the OFAT configuration. The tokens for LayoutLMv3 and ERNIE-Layout include the fine-tuning. *We run Qwen2.5-vision, LayoutLMv3, ERNIE-Layout, and LlaMa3 locally, without API costs.

Models	Tokens	API Cost	STL (F1)	UTL (F1)
GPT-3.5	322K	$0.18	0.777	0.776
GPT-4o	278K	$0.40	0.798	0.788
LLaMA3	239K	$0*	0.784	0.776
GPT-4o-vision	585K	$4.76	0.902	0.897
Qwen2.5-vision	385K	$0*	0.890	0.891
LayoutLMv3	176.1M	$0*	0.603	0.194
ERNIE-Layout	240M	$0*	0.663	0.397

some models enhance document understanding through multimodal learning (OpenAI, 2023; Anil et al., 2023). Structural-aware approaches (Fujitake, 2024; Lee et al., 2022) further improve extraction accuracy, while end-to-end models (Kim et al., 2022) bypass OCR for direct document image processing. Furthermore, some approaches adopt a chat-based paradigm for flexible IE (Xu et al., 2024a), while others (Jiang et al., 2025) enhance document understanding through layout-aware pretraining, advancing LLMs for practical applications.

Strategies for IE from LRDs. Graph-based models (Liu et al., 2019a; Nourbakhsh et al., 2024) enhance relation extraction by capturing textual-visual relationships. A critical factor in layout-rich document (LRD) processing is maintaining the reading order, where Token Path Prediction (TPP)(Zhang et al., 2023) resolves OCR layout ambiguities, and global tagging(Shaojie et al., 2023) mitigates text ordering issues, thereby improving extraction accuracy. For structured data, specific approaches target various document types: table ex-

traction (Jain et al., 2020), form processing (Zhong et al., 2020), and unified OCR, preprocessing, and postprocessing for document IE (Perot et al., 2024). Additionally, a simple yet effective multimodal and multilingual method for parsing semi-structured forms is presented in (Cheng et al., 2025). Despite the variety of models proposed, comparative studies evaluating text-only, multimodal, and layout-aware LLMs remain limited (Van Landeghem et al., 2023; Stanisławek et al., 2021).

Preprocessing, Chunking, Prompting, Postprocessing, and Evaluation Techniques. Chain-of-Thought (CoT) prompting (Wei et al., 2022) enhances reasoning in complex LRD extraction tasks, while diverse prompt-response datasets (Zmigrod et al., 2024a) improve LLM robustness. Instruction-finetuned LLMs have also demonstrated effectiveness in domain-specific applications, such as clinical and biomedical tasks, where zero-shot and few-shot enable adaptive extraction without extensive fine-tuning (Labrak et al., 2024). Post-processing techniques, including text normalization, schema alignment (Fürst et al., 2023), entity resolution (Hwang et al., 2021), and majority voting (Wang et al., 2022a), refine extracted data by correcting OCR and extraction errors.

Despite advances, current studies often focus on isolated components rather than evaluating complete IE pipelines, leading to an incomplete understanding of their interplay. Our findings, quantified and contextualized within a controlled and reproducible framework. Our study enables structured evaluation of IE techniques and systematically analyzes the trade-offs between performance and efficiency across diverse model categories.

6 Conclusion

This paper introduces the design space for IE from LRDs using LLMs, with challenges of data structuring, model engagement, and output refinement, considering the design choices within those challenges. We examine the effects of these choices on the overall performance through one-factor-at-a-time (OFAT) and full-factorial (Brute-Force) exploration of the parameters. We show that well-configured general-purpose LLMs provide an effective alternative to specialized models for IE from LRDs. Our cost-effective OFAT method (2.8% of required computation) generalizes across two datasets (VRDU and FUNSD) with a performance that is just 0.8–1.8% lower than the best-possible configuration that can be found through a full-factorial exploration. This makes our results and method directly applicable to IE practitioners who need to configure their IE pipeline for their dataset. Towards this goal, we have open-sourced LayIE-LLM, our layout-aware IE test suite (`https://github.com/gayecolakoglu/LayIE-LLM`).

Acknowledgments

This work was partially supported by DataGEMS, funded by the European Union's Horizon Europe Research and Innovation Programme, under grant agreement No 101188416.

Limitations

Our experimental evaluation focused on the Visually Rich Document Understanding (VRDU) dataset (Wang et al., 2023c) and the FUNSD dataset (Jaume et al., 2019). VRDU was chosen for its comprehensive OCR outputs, well-defined schema, and diverse template types (Amendment, Short-Form, and Dissemination) that capture complex layouts. It also supports generalization tasks–Single Template Learning (STL), Unseen Template Learning (UTL), and Mixed Template Learning (MTL)–making it suitable for visually rich document understanding. In contrast, FUNSD was selected due to its widespread use in document understanding tasks, particularly for extracting structured information from simpler, form-based layouts. While FUNSD does not share the layout diversity and complexity of VRDU, it serves as a useful baseline for form-centric document analysis. This provides a different perspective by offering insights into scenarios where key-value pairs are organized within relatively simple templates.

One key limitation arises from the differences in schema structure between these datasets. The VRDU dataset has structured templates that facilitate consistent key mapping during post-processing (see Section 3.3), while the FUNSD dataset features dynamic, document-specific key-value pairs, making mapping more challenging. To address this, we plan to enhance our existing schema-mapping step by incorporating context-aware strategies that dynamically adapt to variations and leverage semantic similarity to reduce mismatches.

Another limitation lies in the construction of few-shot prompts (see Appendix A.3). Currently, the LLM generates these prompts by condensing the original OCR data while preserving only text features, aiming to reduce token usage and cost. However, in the tested document, both text and layout features are preserved without modification, embedding spatial structure into text sequences to enhance model understanding. Improving the LLM-generated few-shot prompt process by better integrating spatial features directly into the prompts could enhance the model's performance in complex scenarios.

Our manual investigation (as detailed in Appendix A.1) demonstrated that fuzzy match is more tolerant of valid variations while maintaining high accuracy, as verified through manual evaluation; however, we acknowledge that it may relax correctness criteria and inflate performance metrics. To address this, we plan to extend the evaluation with metrics commonly used in the context of LLMs, such as ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002) scores and their adaptations (Yang et al., 2018). A fair comparison using cost-related metrics is mostly harder to compute due to their changing depending on the scenario. For instance, for Llama3, ERNIE-Layout, and LayoutLMv3, we listed the cost as 0 due to having a free-to-use service; it comes with a cost. Similarly, the cost of computation can be considered from different perspectives, such as the energy usage of a model.

The test suite is currently limited to the steps that we listed in Fig. 2, whereas one could imagine additional steps or factors that affect the performance and may even deliver more satisfactory outcomes. We would like to incorporate additional steps that we learn from the community and incrementally enlarge the design space, and extend the testing capabilities for IE from LRDs.

Another limitation stems from the OFAT strategy used to explore the parameter space. While OFAT enables systematic tuning with minimal computational cost, it inherently fails to detect interaction effects between parameter combinations. To evaluate whether this limitation affected our results, we performed a brute-force search as a benchmark, directly comparing the OFAT outcomes to the best results from the brute-force approach. The comparison showed that OFAT approximated the full-grid results well, indicating that, in our specific scenario, OFAT was sufficient despite its theoretical limitations. We consider this trade-off acceptable for our goal of providing a fast, reproducible tuning method for real-world LLM deployment. Nevertheless, future work may incorporate interaction-aware designs to further uncover parameter synergies.

Last limitation of our study is the variability in performance between the different types of models we tested when handling LRDs. While some models, such as LayoutLMv3 and ERNIE-Layout, are fine-tuned for layout-aware tasks, others, like LLaMA3 and GPT-4o-vision, are optimized for text processing or multi-modal inputs, with their relative strengths and weaknesses varying across the evaluated dimensions. This underscores the need for broader benchmarking with additional models, such as DocLLM, LayoutLLM, and ReLayout, to better understand performance differences.

Ethical Considerations

Large Language Models (LLMs) can contain biases that can have a negative impact on marginalized groups (Gallegos et al., 2024). For the task of information extraction, this could have the impact that uncommon names for people and places are auto-corrected by the LLM to their more common form. In our experiments, we have encountered some instances of this and plan to investigate this further.

References

Rohan Anil et al. 2023. Gemini pro: Integrating multimodal capabilities for enhanced information extraction. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. 2021. Docformer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 993–1003.

Fan Bai, Junmo Kang, Gabriel Stanovsky, Dayne Freitag, Mark Dredze, and Alan Ritter. 2024. Schema-driven information extraction from heterogeneous tables. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10252–10273, Miami, Florida, USA. Association for Computational Linguistics.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Divya Bhadauria, Alejandro Sierra Múnera, and Ralf Krestel. 2024. The effects of data quality on named entity recognition. In *Proceedings of the Ninth Workshop on Noisy and User-generated Text (W-NUT 2024)*, pages 79–88, San Ġiljan, Malta. Association for Computational Linguistics.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. 2019. Scene text visual question answering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4291–4301.

Tsachi Blau, Moshe Kimhi, Yonatan Belinkov, Alexander Bronstein, and Chaim Baskin. 2024. Context-aware prompt tuning: Advancing in-context learning with adversarial methods. *arXiv preprint arXiv:2410.17222*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Xianfu Cheng, Hang Zhang, Jian Yang, Xiang Li, Weixiao Zhou, Fei Liu, Kui Wu, Xiangyuan Guan, Tao Sun, Xianjie Wu, Tongliang Li, and Zhoujun Li. 2025. XFormParser: A simple and effective multimodal multilingual semi-structured form parser. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 606–620, Abu Dhabi, UAE. Association for Computational Linguistics.

Lei Cui, Yiheng Xu, Tengchao Lv, and Furu Wei. 2021. Document ai: Benchmarks, models and applications. *arXiv preprint arXiv:2111.08609*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xin Luna Dong and Divesh Srivastava. 2013. Big data integration. In *2013 IEEE 29th international conference on data engineering (ICDE)*, pages 1245–1248. IEEE.

Masato Fujitake. 2024. LayoutLLM: Large language model instruction tuning for visually rich document understanding. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 10219–10224, Torino, Italia. ELRA and ICCL.

Jonathan Fürst, Mauricio Fadel Argerich, and Bin Cheng. 2023. Versamatch: ontology matching with weak supervision. In *49th Conference on Very Large Data Bases (VLDB), Vancouver, Canada, 28 August-1 September 2023*, volume 16, pages 1305–1318. Association for Computing Machinery.

Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. 2024. Bias and fairness in large language models: A survey. *Computational Linguistics*, pages 1–79.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4083–4091.

Sunghwan Hwang et al. 2021. Entity resolution in heterogeneous data sources: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

Priya Jain et al. 2020. Tabbypdf: Table structure recognition in pdf documents. *arXiv preprint arXiv:2007.12308*.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Zhouqiang Jiang, Bowen Wang, Junhao Chen, and Yuta Nakashima. 2025. ReLayout: Towards real-world document understanding via layout-enhanced pre-training. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3778–3793, Abu Dhabi, UAE. Association for Computational Linguistics.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517. Springer.

Yanis Labrak, Mickael Rouvier, and Richard Dufour. 2024. A zero-shot and few-shot study of instruction-finetuned large language models applied to clinical and biomedical tasks. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2049–2066, Torino, Italia. ELRA and ICCL.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *Preprint*, arXiv:2203.08411.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. 2019a. Graph convolution for multimodal information extraction from visually rich documents. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 32–39, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chuwei Luo, Yufan Shen, Zhaoqing Zhu, Qi Zheng, Zhi Yu, and Cong Yao. 2024a. Layoutllm: Layout instruction tuning with large language models for document understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15630–15640.

Yangyifei Luo, Zhuo Chen, Lingbing Guo, Qian Li, Wenxuan Zeng, Zhixin Cai, and Jianxin Li. 2024b. Asgea: Exploiting logic rules from align-subgraphs for entity alignment. *Preprint*, arXiv:2402.11000.

Margot Mieskes and Stefan Schmunk. 2019. OCR quality and NLP preprocessing. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 102–105, Florence, Italy. Association for Computational Linguistics.

Armineh Nourbakhsh, Zhao Jin, Siddharth Parekh, Sameena Shah, and Carolyn Rose. 2024. AliGATr: Graph-based layout generation for form understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13309–13328, Miami, Florida, USA. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *OpenAI*.

OpenAI. 2024. Hello gpt-4o.

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.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. Cord: a consolidated receipt dataset for post-ocr parsing. In *Workshop on Document Intelligence at NeurIPS 2019*.

Qiming Peng, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Teng Hu, Weichong Yin, Yongfeng Chen, Yin Zhang, et al. 2022. Ernie-layout: Layout knowledge enhanced pre-training for visually-rich document understanding. *arXiv preprint arXiv:2210.06155*.

Vincent Perot, Kai Kang, Florian Luisier, Guolong Su, Xiaoyu Sun, Ramya Sree Boppana, Zilong Wang, Zifeng Wang, Jiaqi Mu, Hao Zhang, Chen-Yu Lee, and Nan Hua. 2024. LMDX: Language model-based document information extraction and localization. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15140–15168, Bangkok, Thailand. Association for Computational Linguistics.

Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. 2021. Going full-tilt boogie on document understanding with text-image-layout transformer. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16*, pages 732–747. Springer.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

He Shaojie, Wang Tianshu, Lu Yaojie, Lin Hongyu, Han Xianpei, Sun Yingfei, and Sun Le. 2023. Document information extraction via global tagging. In *Proceedings of the 22nd Chinese National Conference on Computational Linguistics*, pages 726–735, Harbin, China. Chinese Information Processing Society of China.

Ray Smith. 2007. An overview of the tesseract ocr engine. In *Ninth international conference on document analysis and recognition (ICDAR 2007)*, volume 2, pages 629–633. IEEE.

Tomasz Stanisławek, Filip Graliński, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek. 2021. Kleister: key information extraction datasets involving long documents with complex layouts. In *International Conference on Document Analysis and Recognition*, pages 564–579. Springer.

Antonio Tamayo, Alexander Gelbukh, and Diego Burgos. 2022. NLP-CIC-WFU at SocialDisNER: Disease mention extraction in Spanish tweets using transfer learning and search by propagation. In *Proceedings of The Seventh Workshop on Social Media Mining for Health Applications, Workshop & Shared Task*, pages 19–22, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Zineng Tang, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal. 2023. Unifying vision, text, and layout for universal document processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19254–19264.

Jordy Van Landeghem, Rubèn Tito, Łukasz Borchmann, Michał Pietruszka, Pawel Joziak, Rafal Powalski, Dawid Jurkiewicz, Mickaël Coustaty, Bertrand Anckaert, Ernest Valveny, et al. 2023. Document understanding dataset and evaluation (dude). In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19528–19540.

Dongsheng Wang, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu. 2023a. Docllm: A layout-aware generative language model for multimodal document understanding. *arXiv preprint arXiv:2401.00908*.

Lin Wang et al. 2022a. Majority voting for improved consistency in information extraction tasks. *arXiv preprint arXiv:2207.05214*.

Yanbo J. Wang, Sheng Chen, Hengxing Cai, Wei Wei, Kuo Yan, Zhe Sun, Hui Qin, Yuming Li, and Xiaochen Cai. 2022b. A GlobalPointer based robust approach for information extraction from dialog transcripts. In *Proceedings of the Towards Semi-Supervised and Reinforced Task-Oriented Dialog Systems (SereTOD)*, pages 13–18, Abu Dhabi, Beijing (Hybrid). Association for Computational Linguistics.

Zilong Wang, Yichao Zhou, Wei Wei, Chen-Yu Lee, and Sandeep Tata. 2023b. Vrdu: A benchmark for visually-rich document understanding. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23. ACM.

Zilong Wang, Yichao Zhou, Wei Wei, Chen-Yu Lee, and Sandeep Tata. 2023c. Vrdu: A benchmark for visually-rich document understanding. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5184–5193.

Jason Wei et al. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Jun Xu, Mengshu Sun, Zhiqiang Zhang, and Jun Zhou. 2024a. ChatUIE: Exploring chat-based unified information extraction using large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 3146–3152, Torino, Italia. ELRA and ICCL.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2020a. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1192–1200.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking. *arXiv preprint arXiv:2204.08387*.

Yiheng Xu, Jingjing Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2021. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024b. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

An Yang, Kai Liu, Jing Liu, Yajuan Lyu, and Sujian Li. 2018. Adaptations of rouge and bleu to better evaluate machine reading comprehension task. *arXiv preprint arXiv:1806.03578*.

Chong Zhang, Ya Guo, Yi Tu, Huan Chen, Jinyang Tang, Huijia Zhu, Qi Zhang, and Tao Gui. 2023. Reading order matters: Information extraction from visually-rich documents by token path prediction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13716–13730, Singapore. Association for Computational Linguistics.

Zhenrong Zhang, Jiefeng Ma, Jun Du, Licheng Wang, and Jianshu Zhang. 2022. Multimodal pre-training based on graph attention network for document understanding. *IEEE Transactions on Multimedia*, 25:6743–6755.

Jian Zhong et al. 2020. Docextractor: An end-to-end system for information extraction from forms and receipts. *arXiv preprint arXiv:2012.04573*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Ran Zmigrod, Pranav Shetty, Mathieu Sibue, Zhiqiang Ma, Armineh Nourbakhsh, Xiaomo Liu, and Manuela Veloso. 2024a. "what is the value of templates?" rethinking document information extraction datasets for LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13162–13185, Miami, Florida, USA. Association for Computational Linguistics.

Ran Zmigrod, Dongsheng Wang, Mathieu Sibue, Yulong Pei, Petr Babkin, Ivan Brugere, Xiaomo Liu, Nacho Navarro, Antony Papadimitriou, William Watson, Zhiqiang Ma, Armineh Nourbakhsh, and Sameena Shah. 2024b. Buddie: A business document dataset for multi-task information extraction. *Preprint*, arXiv:2404.04003.

A Appendix

A.1 Discussion & Observations

Effect of Dataset Characteristics on Parameter Selection. The differences in parameter selection between the VRDU (*OCR–Max–Few-shot–0–Cleaned Pred.–Fuzzy Match*) and FUNSD (*Markdown–Small–Few-shot–0–Cleaned Pred.–Fuzzy Match*) datasets stem from their inherent characteristics. VRDU contains long, multi-page documents extracted via OCR, often resulting in noisy and fragmented text, where using the raw OCR output preserves the original structure. In contrast, FUNSD consists of structured, labeled question-answer pairs, making Markdown a suitable input format for key-value pair extraction. These differences shape the parameter selection process, as some models may inherently favor one format over another due to training data characteristics—e.g., GPT-4o may handle OCR data more effectively, while Llama3 shows a preference for Markdown.

Unexpected Impact of Markdown Representation on Model Performance. Markdown, while useful for structuring data, does not always improve performance when applied to noisy and unstructured inputs like those from the VRDU dataset. We observed that converting OCR outputs to Markdown could introduce inconsistencies, such as misaligned headers, broken paragraphs, or flattened tables, affecting data representation quality. In contrast, retaining the raw OCR output often preserved the original structure. For structured datasets like FUNSD, however, Markdown aligned well with the data format, facilitating key-value pair extraction. After experimenting with the IE pipeline on VRDU using purely text input without bounding box locations from OCR, we achieved an overall F1 score of 0.632 (with OCR and Markdown results presented in Table 2). These findings demonstrate that (1) LLMs can leverage structural information from both OCR and Markdown, and (2) their performance varies depending on the input format. These results suggest that the effectiveness of Markdown may depend on both dataset characteristics and model preferences, as previously discussed.

Insights into the Effects of Few-Shot Examples and Chain-of-Thought Prompting. Few-shot prompting with zero examples consistently outperforms other configurations due to the model's ability to generalize without being constrained by specific instances. In VRDU, the variability and noise introduced by OCR, along with the lack of a fixed key-value schema in the FUNSD dataset, make it challenging for the model to generalize from a few specific examples. Few-shot examples often include multiple key-value pairs within a single text that may not appear in other documents, resulting in a mismatch between training and test data. Chain-of-Thought (CoT) prompting, despite its usefulness for reasoning tasks, can overcomplicate straightforward key-value extraction, as guiding the model to think step by step may distract from directly mapping structured content. Moreover, few-shot examples often lack layout information compared to test documents due to the way they were created (see A.3), further contributing to mismatches. Consequently, zero-shot prompting proves more effective by avoiding template-specific biases, maintaining prompt brevity, and enabling better generalization across diverse document structures.

Potential Optimization Bias in Substring and Fuzzy Matching Evaluations. During manual inspection of ground truth and extracted data, we noticed that exact matching often introduces bias by favoring the original annotation, leading to false errors. For example, 'signer_title' in the ground truth is 'general manager', while the input file and extraction have 'general manager, north america'—both correct, yet exact matching would mark the latter as wrong. Similarly, for 'signer_name', the ground truth has ['Catherine Redman-randell'], but the extraction includes ['A.j. Maltrentin', 'Catherine Redman-randell'], which is accurate given the context.

To assess the reliability of substring and fuzzy matching, we manually annotated errors, categorizing them as OCR error, ground truth error, LLM hallucination, additional/wrong information, human error, or incomplete prediction. Substring matching reported zero errors, while fuzzy matching had just one error among 1,730 extractions (see Appendix A.4). These findings support the use of fuzzy matching in scenarios where exact matching would unfairly penalize valid variations.

Selection Criteria for OFAT and Brute-Force Configurations in Performance Evaluation. The best case configurations for each model were determined using a full-factorial (brute-force) exploration, as detailed in Table 8, since the optimal configuration for one model may not gener-

alize well to others. In contrast, for the OFAT configurations, we selected the parameter set that achieved the highest average performance across comparisons—specifically, *OCR–Max–Few-shot–0–Cleaned Pred.–Fuzzy Match*—as shown in Tables 2–5.

While it might seem inconsistent to use OFAT for overall results and brute-force for best/worst cases, the rationale is that OFAT aims to identify generally optimal parameters with minimal effort, whereas the brute-force method is necessary to assess each model's best and worst performance individually for comparison purposes.

A.2 Prompt Generation Details

Prompt Template: Few-shot

```
        ### Examples ###
(<Document>)
{DOCUMENT_REPRESENTATION}
(</Document>)
(<Task>)
{TASK_DESCRIPTION}
{SCHEMA_REPRESENTATION}
(</Task>)
(<Extraction>)
{EXTRACTION}
(</Extraction>)
        ### New Documents ###
(<Document>)
{DOCUMENT_REPRESENTATION}
(</Document>)
(<Task>)
{TASK_DESCRIPTION}
{SCHEMA_REPRESENTATION}
(</Task>)
(<Extraction>)
```

Figure 4: Few-Shot Prompt Structure with 1-shot Example.

Document representation in the example section of the prompt is generated by the LLM, condensing the original OCR data while retaining information relevant to the target schema to reduce token usage and cost. This version integrates only text features. However, in the new example section, both text and layout features are preserved without modification, embedding spatial structure into text sequences to enhance model understanding. This is the only part that differs between the example and new document sections of the prompt. *Task descriptions* provide clear extraction guidelines, specifying what information to retrieve, while *schema representation* defines the expected JSON format to ensure consistency in extracted data. Additionally, CoT prompting includes a *reasoning* component, guiding the model through logical steps to improve accuracy on complex tasks (see Figures 4, 5).

Prompt Template: CoT

```
        ### Examples ###
(<Document>)
{DOCUMENT_REPRESENTATION}
(</Document>)
(<Task>)
{TASK_DESCRIPTION}
{SCHEMA_REPRESENTATION}
(</Task>)
(<Reasoning>)
{REASONING}
(</Reasoning>)
(<Extraction>)
{EXTRACTION}
(</Extraction>)
        ### New Documents ###
(<Document>)
{DOCUMENT_REPRESENTATION}
(</Document>)
(<Task>)
{TASK_DESCRIPTION}
{SCHEMA_REPRESENTATION}
(</Task>)
(<Reasoning>)
{REASONING}
(</Reasoning>)
(<Extraction>)
```

Figure 5: Chain of Thought Prompt Structure with 1-shot Example.

A.3 Tailoring Dataset for our Test-Suite

VRDU dataset includes two benchmarks, each including train samples of 10, 50, 100, and 200 documents. The Registration Form with six entity types used for this project, see Figure 6.

```
{
  "file_date": "",
  "foreign_principle_name": "",
  "registrant_name": "",
  "registration_num": "",
  "signer_name": "",
  "signer_title": ""
}
```

Figure 6: VRDU Registration Form Entities.

The VRDU dataset includes predefined few-shot splits that consist of train, test, and validation sets. These splits contain 10, 50, 100, and 200 training samples, each with three subsets, as shown in Figure 7. The dataset also includes different levels (Lv1: Single, Lv2: Mixed, and Lv3: Unseen Type)

and various template types (Amendment, Dissemination, and Short-Form).

Figure 7: Representation of few_shot_splits from the VRDU dataset.

For each template-level combination, we selected the first JSON file ending in 0 with 10 training samples. Since this training data will be utilized for few-shot and Chain-of-Thought (CoT) prompting, only the first five documents were chosen from the training samples of the selected JSON files. Each level type (STL, UTL) includes template types (Amendment, Dissemination, Short-Form), each with 0, 1, 3, or 5 examples. In STL, these categories use the first document for one-example prompts, the first three for three-example prompts, and all five for five-example prompts. The same structure applies to UTL, with examples specific to its categories. This ensures consistency across template-level combinations while varying the number of examples in the prompt. Figure 8 shows an example of few-shot and CoT examples. This process was repeated for every level, template type, and example count. The example texts were generated using a Large Language Model (LLM), which was instructed to summarize the provided OCR text for the given document while ensuring the inclusion of target schemas and entities.

For the test files, to ensure a fair comparison,

```
few_shot_examples = {
  "STL": {
    "Amendment": {
      0: [],
      1: [
        {
          "text": "This document is an
              amendment to the regis
              ...",
          "entities": {
            "file_date": "1982-10-31",
            "foreign_principle_name": "
                Japan Trade Center...",
            "registrant_name": "PressAid
                 Center",
            "registration_num": "1833",
            "signer_name": "Akira
                Tsutsumi",
            "signer_title": "Director
                General"
          }
        }
      ],
      3: [
        {...},
        {...},
        {...}
      ],
      5: [
        {...},
        {...},
        {...},
        {...},
        {...},
      ]
    },
    "Dissemination": {...
```

Figure 8: VRDU Registration Form Entities.

we selected the first 40 common files from the chosen JSON files within each template type at each level. This means that for Lv1 Amendment and Lv3 Amendment test files, the first common 40 files were selected as test files, and the same strategy was applied for other template types as well. Due to the mixed nature of test files in Lv2, the mixed template type was excluded from this project.

For GPT-3.5, GPT-4o, and LLaMA3-70B, we used few-shot and CoT examples, while GPT-4o-vision used the same test dataset without few-shot examples, following basic instructions. For LayoutLMv3 and ERNIE-Layout, we used the same training and test datasets as the other models but included the entire validation set (300 samples). This consistent setup ensured fair evaluation across all models.

A.4 Success of Evaluation Techniques

We manually reviewed the results of the baseline experiment from GPT-3.5, GPT-4o, and LLaMA3-70B to assess the success of the substring and fuzzy match metrics. The analysis focused on cases where the exact match score was 0 but substring/-fuzzy was 1, highlighting predictions that failed strict matching but were successfully handled in other techniques. This created a dataset to test how well substring and fuzzy matching handle difficult cases. In total, we examined 91 key-value pairs for fuzzy match and 37 for substring match.

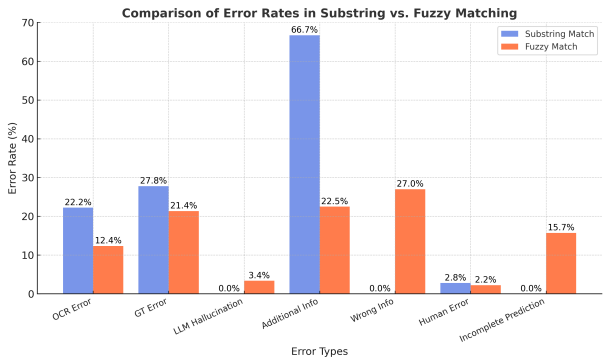

Figure 9: Comparison of error rates in Substring vs. Fuzzy Matching.

Based on our intubation, we identified seven error categories (see Figure 9): OCR errors occur when handwriting is misinterpreted (e.g., "Jim Slattery" as "Jim Slatters"). GT errors arise from incomplete or inaccurate ground truth (e.g., "Om Saudi Arabia 1" instead of "Kingdom Saudi Arabia"). LLM hallucinations involve predictions not present in the OCR or document (e.g., predicting "1992-04-24" instead of "1992-04-21"). Additional info errors include correct predictions with extra information (e.g., "Daniel Manatt Todd" instead of "Daniel Manatt"). Wrong info occurs when LLM selects an incorrect value despite the correct one being available (e.g., predicting "2016-10-31" as the file date instead of "2016-10-08").

Human errors result from physical mistakes, such as crooked or incomplete scans. Incomplete predictions occur when the prediction includes part of the ground truth but misses a segment (e.g., "Japan External Trade Organization" instead of "Japan External Trade Organization Tokyo, Japan"). Figure 9 shows these error types and their rates for both Substring and Fuzzy match categories, where exact matches are labeled as 1.

For evaluating the success of Fuzzy and Substring, we labeled data points as 0 only when the error category is "wrong info"; for other error types, we accepted them and labeled the ground truth as 1. Table 10 presents the performance metrics for Fuzzy and Substring calculated based on these ground truth labels, where exact match is 0 and substring/fuzzy is 1.

Table 10: Performance results of substring and fuzzy evaluation techniques over exact match based on manually annotated data, considering categorized error types.

Data Points	Evaluation Techniques	Precision	Recall	F1
37	Exact Match	0.000	0.000	0.000
	Substring Match	1.000	1.000	1.000
91	Exact Match	0.000	0.000	0.000
	Fuzzy Match	0.984	1.000	0.992

A.5 Additional Experiments with FUNSD Dataset

To demonstrate the generalizability of our techniques, we applied the same steps to the FUNSD dataset as to VRDU.

For the FUNSD dataset, Markdown input consistently outperforms OCR across all models. GPT-3.5 and LLaMA3 show the most significant improvements (+0.100 and +0.103, respectively), while GPT-4o sees a smaller gain (+0.025). On average, Markdown performs better (0.490 vs. 0.414), with lower variability, indicating its effectiveness for structured data in FUNSD (see Table 11).

Table 11: Performance results for different LLMs on the FUNSD dataset with different input types. Baseline configuration in light blue.

Models	Exact Match (F1)	
	OCR	Markdown
GPT-3.5	0.357	0.457 (+0.100)
GPT-4o	0.498	0.523 (+0.025)
LLaMA3	0.387	0.490 (+0.103)
Avg (± stdev.)	0.414 (± 0.074)	0.490 (± 0.033)

Small chunk size yields a slight performance advantage on FUNSD with average F1 of 0.415 over medium (0.414) and max (0.411), likely due to its alignment with FUNSD's short Q&A pairs–preserving structure, reducing context dilution, and focusing models on concise, relevant segments (Table 12).

Zero-shot prompting consistently yields the best performance on the FUNSD dataset, with average

Table 12: Performance results for different LLMs on the FUNSD dataset with different chunk size categories. Baseline configuration in light blue.

Models	Exact Match (F1)		
	Small($\leq$ 1024)	**Medium**($\leq$ 2048)	**Max**($\leq$ 4096)
GPT-3.5	0.354(-0.003)	0.357	0.352(-0.005)
GPT-4o	0.503(+0.005)	0.498	0.494(-0.004)
LLaMA3	0.390(+0.003)	0.387	0.389(+0.002)
Avg ($\pm$ stdev.)	0.415($\pm$ 0.77)	0.414($\pm$ 0.074)	0.411($\pm$ 0.073)

F1 scores of 0.414 (±0.074) for few-shot prompting and 0.404 (±0.109) for CoT. Few-shot prompting is generally more stable than CoT, especially for GPT-3.5 and GPT-4o. Adding examples usually reduces performance, likely due to noise and limited generalization. LLaMA3 remains more stable with CoT compared to other models, but overall, few-shot prompting is better suited for FUNSD's short, structured Q&A format (see Table 13).

Table 13: Different LLMs with different prompt types and example numbers. Baseline configuration is highlighted in light blue.

Models	Exact Match (F1)			
	0	**1**	**3**	**5**
few-shot				
GPT-3.5	0.357	0.289(-0.068)	0.283(-0.074)	0.316(-0.041)
GPT-4o	0.498	0.467(-0.031)	0.399(-0.099)	0.417(-0.081)
LLaMA3	0.387	0.397(+0.010)	0.398(+0.011)	0.393(+0.006)
Avg ($\pm$ stdev.)	0.414($\pm$ 0.074)	0.384($\pm$ 0.089)	0.360($\pm$ 0.66)	0.375($\pm$ 0.052)
CoT				
GPT-3.5	0.327(-0.030)	0.184(-0.173)	0.205(-0.152)	0.218(-0.139)
GPT-4o	0.482(-0.016)	0.396(-0.102)	0.369(-0.129)	0.358(-0.140)
LLaMA3	0.407(+0.020)	0.377(-0.10)	0.360(-0.027)	0.385(-0.002)
Avg ($\pm$ stdev.)	0.404($\pm$ 0.109)	0.290($\pm$ 0.149)	0.287($\pm$ 0.115)	0.288($\pm$ 0.98)

The results indicate that Cleaned Prediction consistently yields better or comparable F1 scores compared to Initial Prediction, with minor improvements due to effective noise reduction, while Mapped Prediction consistently reduces performance due to overgeneralization and key misalignment. This issue arises because the FUNSD dataset lacks a fixed schema, causing key-value pairs to vary dynamically between documents, leading to mapping errors (see Table 14).

The results indicate that Substring Match and Fuzzy Match consistently outperform Exact Match across all models, with average F1 scores of 0.488 (±0.060) and 0.497 (±0.092) respectively, compared to 0.414 (±0.074) for Exact Match. This

Table 14: Different LLMs with different post-processing strategies. Baseline configuration is highlighted in light blue.

Models	Exact Match (F1)		
	Initial Pred.	**Mapped Pred.**	**Cleaned Pred.**
GPT-3.5	0.357	0.311(-0.046)	0.361(+0.004)
GPT-4o	0.498	0.456(-0.042)	0.496(-0.002)
LLaMA3	0.387	0.348(-0.039)	0.393(+0.006)
Avg ($\pm$ stdev.)	0.414($\pm$ 0.074)	0.371($\pm$ 0.075)	0.416($\pm$ 0.070)

improvement is due to the more flexible matching criteria of substring and fuzzy evaluation, which better accommodate minor variations or partial matches in predictions. The performance gain is most pronounced in GPT-4o, highlighting that models producing slightly varied outputs benefit significantly from more tolerant evaluation metrics (see Table 15).

Table 15: Different LLMs with different evaluation techniques. Baseline configuration is highlighted in light blue.

Models	Evaluation Metric (F1)		
	Exact Match	**Substring Match**	**Fuzzy Match**
GPT-3.5	0.357	0.440(+0.083)	0.423(+0.066)
GPT-4o	0.498	0.557(+0.059)	0.601(+0.103)
LLaMA3	0.387	0.469(+0.082)	0.468(+0.081)
Avg ($\pm$ stdev.)	0.414($\pm$ 0.074)	0.488($\pm$ 0.060)	0.497($\pm$ 0.092)

As OFAT, drawing from the underlying 12 experiments, we identified the optimal parameter for each step and each model based on the experimental outcomes (Table 16). Also, we conducted an exhaustive full factorial exploration with 432 configurations to find the best parametrization per LLM (Table 17). Lastly, we find the worst configuration on a per-LLM and per-model basis.

Table 16: OFAT configurations on a per-model basis and corresponding performance results.

Parameter	**GPT-3.5**	**GPT-4o**	**LLaMA3**
Input Type	Markdown	Markdown	Markdown
Chunk Size	Medium	Small	Small
Prompt	Few-Shot	Few-Shot	CoT
Example No.	0	0	0
Output Refin.	Cleaned	Initial	Cleaned
Evaluation	Substring	Fuzzy	Substring

The performance of these different configurations is depicted in Figure 10. The OFAT method achieves an average F1 score of 0.789, marking

Table 17: Brute-Force configurations on a per-model basis and corresponding performance results.

Parameter	GPT-3.5	GPT-4o	LLaMA3
Input Type	Markdown	Markdown	Markdown
Chunk Size	Max	Medium	Small
Prompt	Few-shot	CoT	Few-shot
Example No.	3	0	0
Output Refin.	Cleaned	Cleaned	Cleaned
Evaluation	Fuzzy	Fuzzy	Fuzzy

a substantial gain of 37.5 points over the baseline configuration (0.414). In comparison, the full factorial exploration yields a slightly higher average F1 score of 0.797–only 0.8 points more–while evaluating all 432 configurations. Notably, OFAT achieves this performance by exploring just 12 configurations, requiring approximately 2.8% of the computational effort compared to the full factorial search space. Thus, OFAT captures over 99% of the brute-force method's effectiveness, making it a highly efficient alternative when computational resources are limited. Overall, OFAT continues to offer a compelling balance between performance and efficiency. In comparison, the worst configuration achieves only 0.226 on average, which is approximately 28% of the best configuration.

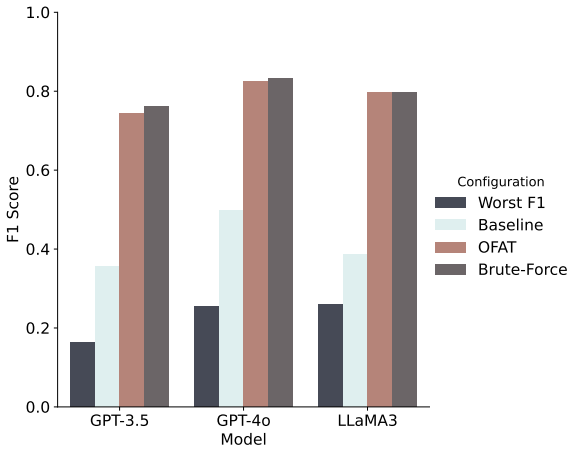

Figure 10: F1 scores of different LLMs across three configurations (Baseline, OFAT, and Brute-Force). Each bar represents the mean F1 score of a model for the corresponding configuration.The OFAT configuration uses a single global parameter set optimized across all models.