WHEN TABLES NEED CONTEXT: QUERY-CONDITIONED TABLE EXTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Most document table extraction systems follow an extract-all paradigm: they reconstruct every table in a document before downstream retrieval or filtering. However, real-world document interaction is often query-driven, where users seek only the tables relevant to a specific information need. We introduce Query-Conditioned Table Extraction (QCTE), a new task that identifies tables satisfying a natural-language condition and reconstructs only the matched tables as structured HTML. We argue that query-conditioned extraction can benefit from document-level context beyond isolated table crops, since captions and surrounding paragraphs provide useful evidence for reliable condition judgment. To this end, we propose PCRC, a framework that combines table-centered context retrieval, progressive evidence incorporation, and reusable KV-cache inference. PCRC progressively introduces contextual evidence only when necessary while sharing visual-textual representations across table selection and reconstruction, substantially reducing redundant multimodal computation. To support this setting, we construct three condition-grounded benchmarks spanning financial, scientific, and visually degraded documents. Across all three datasets, PCRC improves judgment F1 from about 0.73 to 0.90 and composite F1-TEDS by 13–16 points. It requires only 24–29% of the prefill computation of the same inference schedule without cache reuse. These results show that selective document context improves query-conditioned table extraction and can be used efficiently with cache reuse.

1 INTRODUCTION

Tables are important carriers of structured information in financial reports, scientific articles, and regulatory filings, where they are widely used to present statistical data, experimental results, financial indicators, and analytical conclusions. With the development of document intelligence and multimodal large language models, document interaction is increasingly shifting from exhaustive document parsing toward task-oriented information access. In many practical scenarios, users are interested not in all tables contained in a document, but in the subset that satisfies a specific information need, such as tables reporting quarterly revenue changes, particular experimental results, or performance comparisons (Li et al., 2025a; Hein et al., 2025). This motivates a more user-oriented formulation of table extraction: given a natural-language condition, the system should identify the tables that satisfy the condition and extract only those tables. Table 1 compares extract-all table extraction (Zhong et al., 2020; Li et al., 2020) and this query-conditioned formulation.

Table 1: Comparison of extract-all and query-conditioned table extraction.

Aspect	**Extract-all table extraction**	**Query-conditioned table extraction**
Input	Document	Document + natural language condition
Output	All tables in structured form	Matching tables in structured form
Relevance judgment	Not required	Required for table selection

Such query-conditioned table extraction, however, cannot always be reliably performed from an isolated table alone. Tables are often embedded in a broader document narrative, and their surrounding text provides semantic evidence that is not explicitly expressed within the table itself. For example,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

captions and nearby paragraphs may specify what a table reports. Besides, tables with similar visual structures may serve substantially different semantic roles or application contexts, with these distinctions becoming apparent only from their surrounding text. Therefore, query-conditioned table extraction can benefit from jointly considering the table itself and its relevant document context.

While modern vision-language models can in principle consume broader document context, naively introducing such context creates a second challenge: computational redundancy. Condition judgment and table reconstruction may require largely the same table image, OCR content, and surrounding evidence, yet treating them as separate inference stages causes these shared multimodal inputs to be repeatedly encoded. The redundancy becomes more pronounced when additional context is introduced iteratively to resolve uncertain judgments, since each round may recompute an increasingly long input prefix. Because document inputs contain large numbers of visual and textual tokens, such repeated prefill can dominate inference cost. Our goal is therefore to cut the tokens a VLM must process per condition: to identify which contextual evidence is useful for query-conditioned table extraction, and to reuse the computation associated with shared evidence across tables, judgment rounds, and subsequent reconstruction.

To this end, we introduce the Query-Conditioned Table Extraction task and propose PCRC (Progressive Context-Reusable Cache), a unified framework for condition-aware table understanding. PCRC progressively incorporates table-related contextual evidence through table-centered relevance modeling, reducing interference from noisy context. Meanwhile, PCRC organizes table selection and reconstruction as a shared progressive reasoning process, enabling different stages to reuse the same visual-textual contextual representations. Built upon decoder-based vision-language models, PCRC further leverages KV (Key-Value) cache (Li et al., 2024; Yuan et al., 2024) to reuse contextual reasoning states across stages, avoiding repeated multimodal inference over shared input prefixes.

Furthermore, to support condition-aware and document-level contextual reasoning, we construct three new benchmarks: FINCONTAB, PUBCONTAB, and DEGCONTAB, covering financial, scientific, and degraded-document scenarios, respectively. Experimental results show that PCRC improves table selection F_1 from approximately 0.73 to 0.90 and lifts composite F_1-TEDS by 13–16 points. Meanwhile, PCRC reduces repeated prefill computation to 24–29% of the no-cache setting, demonstrating the effectiveness and efficiency of cache-aware contextual reasoning.

The main contributions are summarized as follows:

- We introduce the Query-Conditioned Table Extraction task, which identifies condition-relevant tables and reconstructs them into structured HTML.
- We propose PCRC, a unified framework that enables shared contextual reasoning across table selection and reconstruction through progressive reasoning and cache-aware inference.
- We construct three condition-grounded benchmarks, FINCONTAB, PUBCONTAB, and DEGCONTAB, covering financial, scientific, and degraded-document scenarios.
- Extensive experiments show that PCRC consistently improves table selection F_1 and composite F_1-TEDS over existing methods while cutting repeated prefill computation.[1]

2 RELATED WORK

Table Extraction. Table extraction is usually framed as document digitization: detect table regions and reconstruct every detected table as cells, grids, or HTML. PubTabNet/TEDS and PubTables-1M define common recognition and evaluation protocols (Zhong et al., 2020; Smock et al., 2022), while recent systems improve reconstruction with transformer models and document parsers (Nassar et al., 2022; Peng et al., 2024; Wang et al., 2024a). Our setting differs by making extraction condition-dependent rather than extract-all.

Document and Table-conditioned Reasoning. Document models show that page meaning depends on text, layout, and visual appearance (Huang et al., 2022; Tang et al., 2023; Wang et al., 2024b). Work on document-linked tables further shows that captions, paragraphs, and nearby text often determine how a table should be interpreted (Li et al., 2025b). Table reasoning benchmarks likewise

[1] Our code and dataset are available at `https://anonymous.4open.science/r/ICLR_2027_pcrc-D410/`.

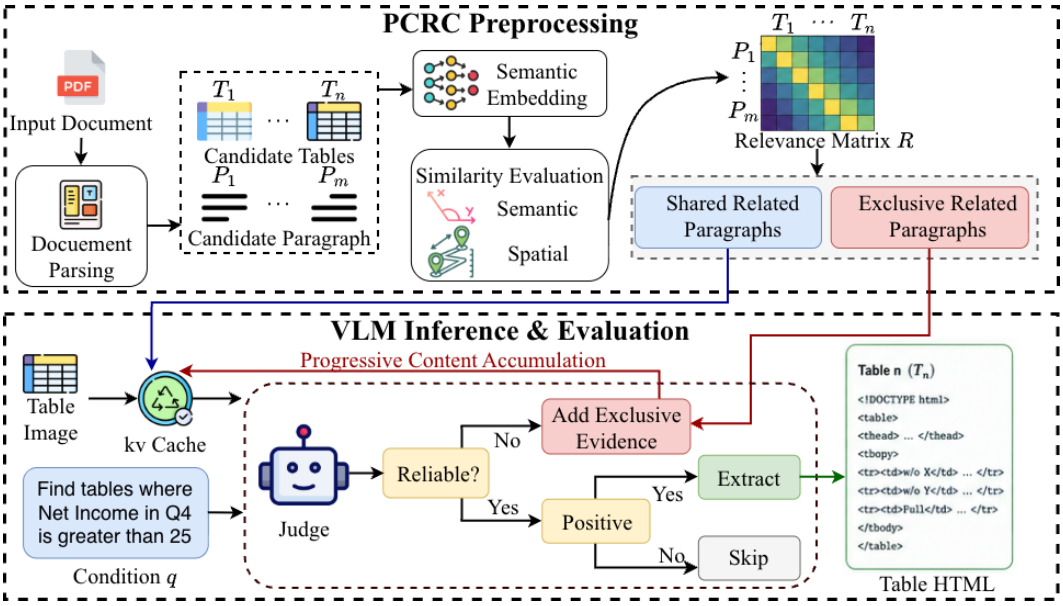


Figure 1: The overall workflow of PCRC

require lookup, verification, numerical reasoning, and multi-step evidence use (Chen et al., 2020; 2021; Zhao et al., 2022; Wu et al., 2025). These directions motivate condition-grounded extraction with related paragraphs.

Query-Conditioned Table and Dataset Discovery. A related line of work conditions table or dataset retrieval on user queries. DSEBench ranks datasets using keyword queries, example datasets, and metadata/content fields (Shi et al., 2025); nlcTables refines table discovery using natural-language requirements over structured candidate tables (Cui et al., 2025b); and UniDisc supports unified discovery over data lakes across natural-language and table queries (Wang et al., 2026). Unlike these methods, which retrieve pre-existing structured tables or datasets, our setting operates on rendered documents, jointly selecting condition-relevant tables and reconstructing them as HTML from visual inputs.

Cache-Aware Inference. Document-context VLM inference is expensive because prefill computes KV states for both visual and textual tokens. Efficient long-context inference therefore studies cache eviction, compression, and reuse (Xiao et al., 2024; Zhang et al., 2023; Li et al., 2024; Yuan et al., 2024), and RAG systems precompute or blend KV caches for retrieved chunks (Jin et al., 2025; Yao et al., 2025). Serving engines reuse the states of prefixes that recur across requests and keep or evict them by recency (Kwon et al., 2023). PCRC instead fixes the roles: document-level shared evidence forms a head that table-specific content never displaces, each table's evidence occupies a tail dropped once its decision is made, and the judgment states are carried into extraction.

3 METHODOLOGY

The overall workflow of PCRC (**P**rogressive **C**ontext-**R**eusable **C**ache) is illustrated in Figure 1. Given a document D and a natural-language condition q, the goal is to identify the tables that satisfy q and reconstruct only those tables as structured HTML. PCRC consists of two main stages: **PCRC Preprocessing** and **VLM Inference & Evaluation**. In the preprocessing stage, PCRC constructs a *Table-Centered Relevance Index* (Section 3.1) over surrounding paragraphs and organizes them into document-level shared evidence and table-specific exclusive evidence. In the inference stage, candidate tables are judged sequentially under the user condition; shared evidence is reused across tables with *Progressive Context-Reusable Cache* (Section 3.2), while exclusive evidence is progressively introduced when additional context is needed. Tables with positive judgments are then reconstructed as HTML using the same cached visual-textual states.

We use an off-the-shelf document parser to identify candidate table regions and surrounding paragraph blocks. Then we apply OCR to obtain their textual content. Modern general-purpose layout detectors already provide strong off-the-shelf performance: for example, PP-DocLayout-L reports 90.4% mAP@0.5 over 23 region categories (Sun et al., 2025). We therefore treat layout detection and OCR as replaceable front-end components rather than contributions of this work; a stronger parser can be substituted without changing the PCRC formulation or inference procedure. We denote the detected candidate tables by $\mathcal{T}(D) = \{T_1, \ldots, T_n\}$ and the surrounding text blocks by $\mathcal{P}(D) = \{P_1, \ldots, P_m\}$. The OCR associated with each table is used only as an unstructured text sequence for relevance modeling and VLM evidence, rather than as a structured table representation.

3.1 Table-Centered Relevance Index

PCRC preprocessing builds a table-centered relevance index over the paragraphs based on a paragraph-table score $r(P_j, T_i)$, which combines semantic similarity with spatial proximity. For each paragraph-table pair, we define their relevance as:

$$r(P_j, T_i) = s_{\text{sem}}(P_j, T_i) \cdot s_{\text{spa}}(P_j, T_i), \tag{1}$$

where s_{sem} measures semantic relatedness in an embedding space and s_{spa} is a layout-based distance correction. The product form expresses a mutual-agreement principle: a paragraph should be useful only when it is semantically related to the table and spatially plausible as its document context. This mirrors multiplicative appearance/geometric fusion in relation networks (Hu et al., 2018) rather than the additive layout treatment of document pretraining models (Xu et al., 2020; Tang et al., 2023; Wang et al., 2024b). Both factors lie in $[0, 1]$, so the product is bounded and scale-compatible without further calibration: a weak semantic match drives the score toward zero, and the spatial factor scales it within $[1/e, 1]$, whereas a sum would let one strong signal mask a missing one. Appendix A.4.4 verifies empirically that neither factor alone is sufficient, and Appendix A.4.6 shows that the two factors are also balanced in practice, neither one dominating the product.

Semantic Similarity. We use a pretrained encoder ϕ to map the OCR text of each paragraph and table into a shared semantic space. Let $\mathbf{p}_j = \phi(\text{OCR}(P_j))$ and $\mathbf{t}_i = \phi(\text{OCR}(T_i))$ denote the paragraph and table embeddings, respectively. The semantic similarity is $s_{\text{sem}}(P_j, T_i) = \max\{0, \cos(\mathbf{p}_j, \mathbf{t}_i)\}$.

Spatial Proximity. Semantic similarity alone ignores how documents attach explanations to visual objects. We therefore add a spatial correction from the rendered layout. Let $c(X)$ be the center of the bounding box of component X, and let $d_{ij} = \|c(P_j) - c(T_i)\|$, normalized within each document as $\hat{d}_{ij} = d_{ij}/d_{\max} \in [0, 1]$. We convert distance into a decreasing locality score $s_{\text{spa}}(P_j, T_i) = \exp(-\hat{d}_{ij})$; the normalization keeps the spatial factor on the same scale as the semantic one (Appendix A.4.6). Nearby paragraphs thus keep their semantic score while distant ones are down-weighted unless that signal stays competitive; exponential locality functions are a standard way to encode such spatial priors (Yang et al., 2018; Guo et al., 2019).

Relevance Matrix. Applying Eq. 1 to every paragraph-table pair yields a relevance matrix $R = [r(P_j, T_i)]$. PCRC preprocessing converts this matrix into *shared related paragraphs* $\mathcal{S}(D)$ and *exclusive related paragraphs* $\mathcal{E}(T_i)$:

$$\mathcal{I}(D) = (\mathcal{S}(D), \{\mathcal{E}(T_i)\}_{i=1}^{n}), \tag{2}$$

where "shared" and "exclusive" refer only to how a paragraph is related to tables in the document.

A shared related paragraph is a paragraph that has high relevance to all or most tables in the document. For each paragraph, we average its relevance over all tables, $g(P_j) = \frac{1}{n}\sum_{i=1}^{n} r(P_j, T_i)$. Paragraphs whose document-level score exceeds a threshold τ_{shared} are selected into $\mathcal{S}(D)$, optionally truncated to a budget K_{shared}. These paragraphs can serve as common reference information when judging multiple tables under the same condition.

An exclusive related paragraph is highly relevant to one or a small number of tables, but is not broadly relevant to the document's table set. For each table, we select its ordered exclusive related paragraphs as:

$$\mathcal{E}(T_i) = \text{TopK}_{K_{\text{excl}}}\{P_j \notin \mathcal{S}(D) : r(P_j, T_i) \geq \tau_{\text{excl}}\}. \tag{3}$$

The ordering follows decreasing relevance, producing $(e_1^{(i)}, e_2^{(i)}, \ldots)$. These paragraphs provide table-specific supporting evidence for judging whether the associated table satisfies the condition.

Algorithm 1 PCRC inference for a document-condition pair

Require: shared cache C_{head}, tables $\mathcal{T}(D)$, exclusive sets $\mathcal{E}(T_i)$
1: **for** each table $T_i \in \mathcal{T}(D)$ **do**
2: $C \leftarrow C_{\text{head}} \oplus [\text{image}(T_i); \text{ocr}(T_i)]$; $(a, c) \leftarrow \text{Judge}(C)$; $k \leftarrow 1$
3: **while** $c \neq$ RELIABLE and $k \leq |\mathcal{E}(T_i)|$ **do**
4: drop the KV states of the request and the uncertain answer; $C \leftarrow C \oplus e_k^{(i)}$; $(a, c) \leftarrow \text{Judge}(C)$; $k \leftarrow k + 1$
5: **end while**
6: **if** $c \neq$ RELIABLE **then** $a \leftarrow \text{Judge}(C \oplus$ forced yes/no prompt$)$
7: **if** $a =$ YES **then** output $\text{Extract}(C \oplus$ extraction inputs$)$ **else** skip T_i
8: discard the table-specific tail
9: **end for**

3.2 PROGRESSIVE CONTEXT-REUSABLE CACHE

Many table-condition judgments need surrounding paragraphs as evidence, but feeding every document image and paragraph to the VLM at once would make prefill prohibitive. PCRC therefore processes candidate tables sequentially, giving the VLM the cropped table image, table OCR, and the paragraphs chosen by the relevance index, and reuses KV states at two levels. Across tables, shared related paragraphs are cached once as document-level evidence. Within a table, exclusive related paragraphs are appended progressively while previously injected evidence stays in the cache, so each round encodes only the newly added evidence instead of the full prompt.

Fixed Head and Replaceable Tail. PCRC organizes inference as a cache frame with a fixed head and a table-specific tail. For a document-condition pair (D, q), the fixed head is defined as

$$\Pi_{\text{head}}(D, q) = [\iota_{\text{sys}};\ \iota_{\text{judge}};\ q;\ \mathcal{S}(D)], \tag{4}$$

where ι_{sys} is the system instruction, ι_{judge} is the judgment policy, q is the user condition, and $\mathcal{S}(D)$ is the set of shared related paragraphs. The shared related paragraphs $\mathcal{S}(D)$ are placed in this head because they can support judgments for many tables. We prefill Π_{head} once and store its KV cache C_{head}. For the first table this is a normal prefill cost; for subsequent tables, the same shared evidence is available without recomputing its key/value states.

The tail stores the KV states that are specific to the currently visited table. It may contain the cropped table image, the table OCR, progressively added exclusive related paragraphs, and, after a positive decision, the sample table image and sample HTML used to guide extraction. When PCRC moves from T_i to T_{i+1}, this tail is discarded and replaced by the new table's tail. Within the same table, the tail is not rebuilt from scratch: it grows by appending newly needed exclusive evidence.

With this cache structure, PCRC processes the document as a sequence of table decisions. Each table starts from the shared head and a fresh table-specific tail; exclusive related paragraphs are appended only when the current judgment is not reliable. A reliable positive decision triggers extraction with the sample HTML reference, while a reliable negative decision stops the current table.

The reliability signal is produced by the model itself rather than by an external classifier: the judging prompt permits a third answer, `uncertain`, alongside `yes` and `no`. Whenever the model answers `uncertain`, PCRC appends the next exclusive paragraph and re-judges. This repeats until the answer becomes definite or the exclusive set is exhausted; after the last exclusive paragraph has been appended, an explicit prompt forces the model to commit to a definite yes/no answer, so every table receives a decision.

Cache Reuse Effect. Algorithm 1 makes two reuse boundaries explicit. Across tables, the table-specific tail is removed while C_{head} is retained. Within a table, each additional round computes KV states only for the newly inserted exclusive paragraph; the table image, table OCR, shared paragraphs, and earlier exclusive paragraphs remain available through the existing cache. When a table is accepted, extraction is appended to the same cache: the KV states already computed for judgment, including the table image tokens, table OCR, and related paragraphs, are carried forward to HTML generation rather than recomputed. Conversely, reliability-gated stopping prevents later exclusive paragraphs from being prefetched once the available evidence is already sufficient.

3.3 EVALUATION CRITERIA

We evaluate the task in three layers: judgment quality, extraction quality, and composite quality.

Judgment Quality. For each (D, q, T_i) over the candidates aligned to source tables (Appendix A.3), the prediction is classified as TP, FP, FN, or TN with respect to the positive class. We report precision, recall, and F_1 for table selection, micro-averaged: all table–condition pairs are pooled into one confusion matrix.

Extraction Quality. Extraction is scored only on true positives. False positives are already penalized by precision, and false negatives by recall. On TP tables we use TEDS, the tree-edit similarity introduced with PubTabNet (Zhong et al., 2020), which captures the fidelity of the predicted HTML tree against the reference.

Composite Quality. To avoid rewarding systems that judge well but extract poorly, we multiply judgment precision and recall by the cell-count weighted average extraction quality on true-positive tables, measured with TEDS. This yields composite precision and recall, and we report their harmonic mean F_1. Composite quality is high only when both relevance judgment and table reconstruction are strong.

4 DATASET

We construct three datasets for *conditioned table extraction*. Each example contains a multi-page document image sequence, structured table HTML, surrounding paragraph annotations, and `yes`/`no` labels for every condition–table pair. Systems identify matching tables and return only their HTML (full schema: Appendix A.1.1).

Dataset scale and table characteristics. FINCONTAB fully reuses financial pages from FINTABNET, and PUBCONTAB fully reuses scientific pages from PubTables-style sources. Their table styles and document, page, and table-scale properties follow the source benchmarks (Zheng et al., 2021; Smock et al., 2022). Table 2 summarizes the domains and scales of the three condition-grounded datasets. Each document is paired with three conditions, and each condition is labelled for every table of its document, so positives are a minority: the label-level positive-to-negative ratio is roughly 1:20 on FINCONTAB and 1:15 on DEGCONTAB, and about 1:1 on PUBCONTAB, whose documents contain few tables. The five condition categories (C1–C5) are targeted at approximately 20% each by construction.

Table 2: Summary of the three condition-grounded datasets. Pos:Neg is the ratio of positive to negative condition–table labels.

Dataset	Domain	# Docs	# Tables	# Conditions	Pos:Neg
FINCONTAB	Financial	4,576	112,887	13,728	1:20
PUBCONTAB	Scientific	401,661	947,642	1,204,983	1:1
DEGCONTAB	Degraded / hard-case	23,000	386,000	69,000	1:15

DEGCONTAB challenges judgment and extraction with multi-level headers, varied ruling lines and borders, and watermarks, stains, and stamps. Degradation weakens table-crop reliability; complex rules and headers obscure row/column scope and header dependencies. Scale, table characteristics, splits, and the document-synthesis prompt are in Appendices A.1.3 and A.1.4.

Condition categories. The original sources provide table images and official HTML but no document-level conditions. All three datasets share five added condition categories: C1 lookup and filter, C2 numerical reasoning, C3 multi-constraint conjunction, C4 multi-step reasoning, and C5 structural conditions. These cover recurring abilities in table understanding, including verification and hierarchical reasoning (Chen et al., 2020; Cheng et al., 2022). Conditions and their initial table labels are proposed by Gemini 3.5 Flash (Google DeepMind, 2026) and filtered automatically; a stratified 5% sample of conditions is checked by human annotators against the official table HTML, covering every table label of each sampled condition, with 20% of the sample labelled independently twice; automatic and human labels agree on 98.5% of the checked pairs with a binary human label. See Table 7 and Appendices A.1.2–A.1.7 for details.

5 EXPERIMENT

Section 5.2 asks whether the right document context makes judgment and reconstruction more accurate. Section 5.3 asks whether a reusable cache keeps that context affordable.

5.1 EXPERIMENTAL SETUP

Datasets. We evaluate on three condition-grounded table-extraction datasets, FINCONTAB, PUBCONTAB, and DEGCONTAB. Section 4 describes them. Since the compared methods require no task-specific training, we do not use the train/dev/test split labels for model fitting, and all experiments in the main text and the appendix are run on each dataset in full, subject only to the exclusions of Appendix A.3; the wall-clock timing of Table 16 uses a fixed 20-document sample per dataset. Every method and variant consumes the same parsed inputs from the upstream detector and OCR, so the comparison isolates contextual evidence selection, progressive inference, and cache reuse.

Upstream parsing and relevance encoder. Candidate table regions and paragraph blocks come from PP-Structure of the PaddleOCR toolkit (Cui et al., 2025a; Sun et al., 2025), used off the shelf. The FINCONTAB and DEGCONTAB PDFs carry a text layer, read with PyMuPDF instead of optical recognition; on DEGCONTAB, text under the rendered watermarks, stains, and stamps is masked, so degraded regions are observed only through the image. The encoder ϕ of Equation 1 is NV-Embed (Lee et al., 2025), applied to paragraph and table OCR without fine-tuning.

Methods under comparison. We compare PCRC with a single-shot Qwen baseline, a no-cache variant of PCRC, and six modular pipelines that combine a table extractor with a separate judge.

Qwen-based methods. **(a) Qwen** is a single-shot Qwen2.5-VL-7B-Instruct (Bai et al., 2025) that takes the table image, table OCR, condition, and a one-shot sample table image with its reference HTML as a formatting demonstration, decides whether the table satisfies the condition and, if so, extracts it, returning either `no` or the table HTML in one decoding pass. The same Qwen2.5-VL-7B checkpoint is reused in all Qwen-based methods below, and we refer to it simply as *Qwen*. **(b) PCRC** is our proposed method (Section 3.2), using the same Qwen under the cache-frame inference contract: a fixed head holds the condition and shared related paragraphs. Exclusive related paragraphs are appended progressively, and the KV cache is reused across tables and rounds. **(c) PCRC w/o cache** follows PCRC's inference flow with KV reuse disabled, so every table and progressive round recomputes the full prefix.

Modular pipelines. We also compare six two-stage modular pipelines under the $X + Y$ convention, where X runs first and Y consumes its output. The modules have fixed roles within these pipelines. In the extract-then-judge pipelines, Qwen is run as a VLM judge that takes the condition, the table image, and the upstream HTML and returns `yes`/`no`, and TAPAS (Herzig et al., 2020) is a non-VLM judge that operates on the structured table derived from the upstream HTML. TATR (Smock et al., 2022) and MinerU (Wang et al., 2024a) are used only as non-VLM HTML extractors; TATR is the Table Transformer extractor, and MinerU its non-VLM parsing pipeline rather than the VL variant.

Under this convention, **Qwen+TATR** and **Qwen+MinerU** are judge-then-extract pipelines that keep the single-shot Qwen judgment (a table counts as `yes` when single-shot Qwen returns HTML rather than `no`) and run TATR or MinerU on the accepted tables, so they differ from Qwen only in the extractor. The remaining four pipelines are extract-then-judge pipelines: **TATR+TAPAS**, **TATR+Qwen**, **MinerU+Qwen**, and **MinerU+TAPAS**. TATR or MinerU first extracts HTML for every candidate table, and the downstream judge then decides whether the extracted table satisfies the condition.

Evaluation protocol. Following Section 3.3, judgment quality is micro precision, recall, and F_1 on the positive class; composite quality forms $P_c = P_{\text{judge}} \cdot \text{EQ}$ and $R_c = R_{\text{judge}} \cdot \text{EQ}$, where EQ is cell-count weighted TEDS extraction quality on true positives, and reports their harmonic mean F_1; compute cost is per-condition prefill, decode, and total tokens plus the recomputation ratio against a no-cache reference. Quality is scored on a fixed set of upstream candidates uniquely aligned to the source tables and shared by all methods, hence conditional on that set rather than on end-to-end detection, whereas cost counts all computation on the original candidate set (Appendix A.3; implementation and prompts: Appendices A.4.1 and A.2).

Table 3: Composite quality, computed with TEDS as the extraction metric. FIN, PUB, and DEG denote FINCONTAB, PUBCONTAB, and DEGCONTAB, respectively. **Best** per dataset per column are highlighted in red and second-best results are highlighted in green.

	FIN			PUB			DEG		
Method	P_c	R_c	F_1	P_c	R_c	F_1	P_c	R_c	F_1
Qwen+TATR	0.294	0.354	0.322	0.587	0.772	0.667	0.371	0.408	0.389
Qwen+MinerU	0.481	0.579	0.525	0.467	0.614	0.531	0.548	0.602	0.574
TATR+TAPAS	0.021	0.339	0.040	0.598	0.760	0.669	0.025	0.383	0.046
TATR+Qwen	0.289	0.286	0.287	0.619	**0.777**	0.689	0.338	0.325	0.331
MinerU+Qwen	0.406	0.528	0.459	0.454	0.625	0.526	0.462	0.566	0.509
MinerU+TAPAS	0.044	0.377	0.080	0.361	0.521	0.427	0.033	0.501	0.062
Qwen	0.547	0.658	0.597	0.529	0.696	0.601	0.615	0.676	0.644
PCRC (ours)	**0.713**	**0.734**	**0.723**	**0.741**	0.771	**0.756**	**0.791**	**0.797**	**0.794**
PCRC w/o cache	**0.713**	**0.734**	**0.723**	**0.741**	0.771	**0.756**	**0.791**	**0.797**	**0.794**

5.2 QUALITY COMPARISON

Because the task is complete only when a system both identifies the relevant tables and reconstructs them faithfully, this subsection evaluates the two outputs jointly, as a composite precision/recall pair and its F_1; a judgment-only view is reported in Appendix A.4.2.

Overall results. All figures below are composite scores read from Table 3. PCRC is the strongest method on all three datasets, lifting composite F_1 over single-shot Qwen from $0.597/0.601/0.644$ to $0.723/0.756/0.794$ on FINCONTAB/PUBCONTAB/DEGCONTAB; on FINCONTAB and DEGCONTAB it also leads every non-PCRC method on all three composite columns. PCRC and PCRC w/o cache coincide in every column, confirming that KV-cache reuse leaves output quality unchanged. PCRC's lead over Qwen is reproduced on the human-checked label subset (Table 12).

PCRC versus single-shot Qwen. Both methods judge from the same table image and OCR; the related paragraphs PCRC adds lift composite precision/recall from $0.547/0.658$ to $0.713/0.734$ on FINCONTAB and from $0.615/0.676$ to $0.791/0.797$ on DEGCONTAB.

VLM extraction versus non-VLM extraction. Qwen, Qwen+TATR, and Qwen+MinerU share the same Qwen judgments, so their differences isolate the extraction module. Replacing Qwen's direct HTML generation with TATR or MinerU lowers composite F_1 from 0.597 to 0.322 and 0.525 on FINCONTAB, and from 0.644 to 0.389 and 0.574 on DEGCONTAB. With judgment held fixed, direct VLM generation is therefore the stronger extractor on both out-of-domain datasets.

PUBCONTAB is the expected exception: Qwen+TATR reaches $F_1 = 0.667$ against Qwen's 0.601, PCRC's margin over the best modular pipeline is narrowest (0.756 against TATR+Qwen's 0.689), and PCRC trails on composite recall (0.771 against 0.777), because TATR was pretrained on PubTables-1M, the source family of PUBCONTAB (Smock et al., 2022); outside that domain PCRC reaches 0.723 and 0.794 against TATR+Qwen's 0.287 and 0.331.

5.3 COMPUTATION COST

Compute is measured as the tokens the VLM processes; Qwen, PCRC, and PCRC w/o cache share the same backbone, so their token counts are directly comparable. Per condition, we sum all forward calls, judgment and HTML generation included, and average over conditions. Table 4 reports prefill, decode, total tokens, and the recomputation ratio $\text{R.R.} = \sum P_{\text{actual}} / \sum P_{\text{no-cache}}$, whose denominator is the prefill of the same call sequence, with the same inputs and outputs, when every call re-encodes its full prefix; lower is better, and Qwen has no cache to reuse.

Figure 2 combines the composite scores and prefill costs with the context-selection variants in Appendix A.4.4, including the no-gating ablation, which consumes all selected exclusive paragraphs without early stopping.

KV reuse cuts prefill cost by 3.4–4.1×. PCRC executes only 28.3%, 29.0%, and 24.4% of the no-cache prefill on FINCONTAB, PUBCONTAB, and DEGCONTAB, saving 83.5k, 32.8k, and 60.8k

Table 4: Per-condition token consumption. Pre., Dec., and Tot. are prefill, decode, and total tokens in thousands; R.R. is the recomputation ratio in percent. FIN, PUB, and DEG denote FINCONTAB, PUBCONTAB, and DEGCONTAB. Best between PCRC and PCRC w/o cache on Pre., Tot., and R.R. is in **bold**.

	FIN				PUB				DEG			
Method	Pre.	Dec.	Tot.	R.R.	Pre.	Dec.	Tot.	R.R.	Pre.	Dec.	Tot.	R.R.
Qwen	27.73	1.19	28.92	—	9.46	1.62	11.08	—	13.85	0.72	14.57	—
PCRC	**32.95**	3.57	**36.52**	**28.29**	**13.43**	3.09	**16.52**	**29.02**	**19.57**	3.78	**23.35**	**24.35**
PCRC w/o cache	116.47	3.57	120.04	100.0	46.27	3.09	49.36	100.0	80.36	3.78	84.14	100.0
PCRC schedule + generic prefix cache	33.38	3.57	36.95	28.66	13.47	3.09	16.56	29.13	20.11	3.78	23.89	25.01

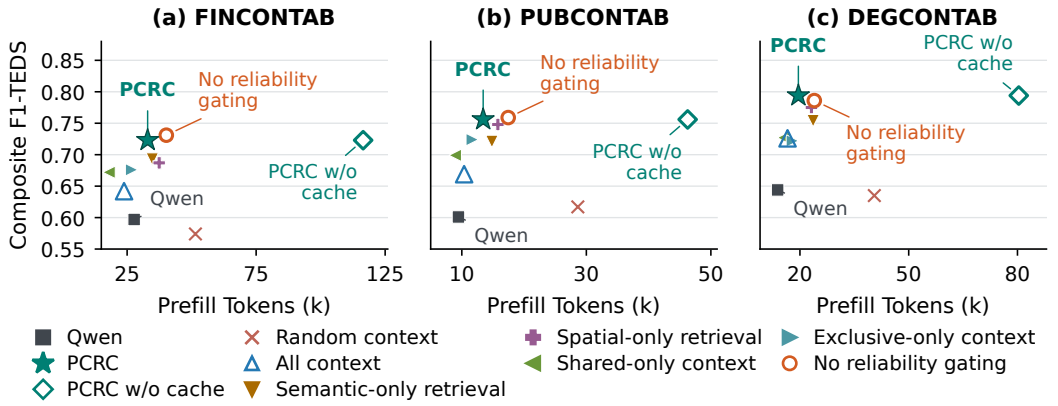


Figure 2: Composite F_1-TEDS versus per-condition prefill tokens (k). PCRC improves quality over Qwen while using substantially fewer prefill tokens than PCRC w/o cache.

prefill tokens per condition. A generic block-level prefix cache replaying the same schedule recomputes 25–29%, within 0.7 points of PCRC (Appendix A.4.5); On the same engine, the progressive schedule is 3–5% faster than the single-judgment protocol on a 60-condition sample (Appendix A.4.5, Table 16). The saving lies in prefill, cache reuse being a prefill-side optimization.

The overhead over single-shot Qwen is modest relative to the quality gain. PCRC adds only $+5.2$k, $+4.0$k, and $+5.7$k prefill tokens over Qwen on FINCONTAB, PUBCONTAB, and DEGCONTAB, and its total token use is 1.3–1.6$\times$ that of Qwen while remaining about $3\times$ fewer tokens than the no-cache schedule. This buys the $+17$ pp judgment F_1 gain in Appendix A.4.2 and the $+13$–16 pp composite gain above. Prefill dominates decode because inputs carry table-image patch tokens. Appendix A.4.3 separates context selection from early stopping and gives the per-category breakdown; Appendix A.4.7 reports a second open-weight backbone and frontier single-shot baselines, and Appendix A.4.8 an extraction-only context ablation.

6 CONCLUSION

We introduced Query-Conditioned Table Extraction, which identifies tables satisfying a natural-language condition and reconstructs only those tables as structured HTML, and proposed PCRC, a Progressive Context-Reusable Cache framework that combines table-centered evidence ranking, progressive context incorporation, and KV-cache reuse. Across three condition-grounded benchmarks spanning financial, scientific, and visually degraded documents, PCRC improves condition judgment and composite extraction quality while substantially reducing repeated prefill computation, highlighting the value of document-level context and computation-aware inference.

Our study focuses on query-conditioned table extraction with paragraph-level document context and inference-time cache reuse. However, extending this setting to broader document domains and languages remains open. Stronger relevance models and task-specific VLM adaptation are also worth exploring.

AI USE STATEMENT

We used generative AI to assist with dataset construction. Gemini 3.5 Flash generated the text and tables for DEGCONTAB. Sampled documents were checked manually (Appendix A.1.3). The same model proposed conditions and initial table labels for all three datasets using the reference HTML. These outputs were filtered automatically. Human annotators then checked a stratified 5% sample (Appendix A.1.6).

We also used generative AI to draft sections of the paper and improve the wording. The authors reviewed and revised all AI-assisted manuscript text. We take full responsibility for the final paper and datasets.

ETHICS STATEMENT

FINCONTAB and PUBCONTAB reuse publicly available open-access sources under permissive licenses; DEGCONTAB documents and all conditions are synthesized with Gemini 3.5 Flash as described in Appendix A.1. The resulting datasets contain only public or synthetic document content and metadata, without personal or sensitive information. Label checking was performed by five unpaid volunteer annotators from the authors' group acting as technical validators; no data about them or any other person was collected, and no human subjects are involved.

REPRODUCIBILITY STATEMENT

We provide the implementation and dataset through the anonymous repository. Dataset construction, prompts, label validation, evaluation protocols, hyperparameters, and implementation details are documented in Appendices (A.1– A.4). All reported baselines use the same frozen candidate set and evaluation protocol described in Appendix A.3.

REFERENCES

Alibaba Cloud. Qwen3-VL-Plus. Alibaba Cloud Model Studio, 2026. URL https://docs.modelstudio.console.alibabacloud.com/en/model-studio/qwen3-vl-plus. Model documentation; qwen3-vl-plus is functionally equivalent to the qwen3-vl-plus-2025-12-19 snapshot.

Anthropic. System card: Claude opus 4.5, November 2025. URL https://www-cdn.anthropic.com/bf10f64990cfda0ba858290be7b8cc6317685f47.pdf.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. TabFact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkeJRhNYDH.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R. Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.emnlp-main.300.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. HiTab: A hierarchical table dataset for question answering and natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1094–1110. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.acl-long.78.

Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. PaddleOCR 3.0 technical report. *arXiv preprint arXiv:2507.05595*, 2025a. URL https://arxiv.org/abs/2507.05595.

Lingxi Cui, Huan Li, Ke Chen, Lidan Shou, and Gang Chen. nlcTables: A dataset for marrying natural language conditions with table discovery. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025b. URL https://doi.org/10.1145/3726302.3730296.

Google DeepMind. Gemini 3.5 flash: Model card, May 2026. URL https://deepmind.google/models/model-cards/gemini-3-5-flash/. Published May 19, 2026.

Maosheng Guo, Yu Zhang, and Ting Liu. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 6489–6496, 2019.

David Hein, Alana Christie, Michael Holcomb, Bingqing Xie, A. J. Jain, Joseph Vento, Neil Rakheja, Ameer Hamza Shakur, Scott Christley, Lindsay G. Cowell, James Brugarolas, Andrew Jamieson, and Payal Kapur. Prompts to table: Specification and iterative refinement for clinical information extraction with large language models. *medRxiv*, 2025. doi: 10.1101/2025.02.11.25322107. URL https://doi.org/10.1101/2025.02.11.25322107.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4320–4333. Association for Computational Linguistics, 2020. URL https://aclanthology.org/2020.acl-main.398.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14409–14428, 2023.

Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 3588–3597. IEEE, 2018.

Yongshuai Huang, Ning Lu, Dapeng Chen, Yibo Li, Zecheng Xie, Shenggao Zhu, Liangcai Gao, and Wei Peng. Improving table structure recognition with visual-alignment sequential coordinate modeling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11134–11143, 2023.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. LayoutLMv3: Pre-training for document AI with unified text and image masking. In *ACM International Conference on Multimedia*, 2022.

Chao Jin, Zili Zhang, Xuanlin Jiang, Fangyue Liu, Shufan Liu, Xuanzhe Liu, and Xin Jin. RAGCache: Efficient knowledge caching for retrieval-augmented generation. *ACM Transactions on Computer Systems*, 44(1):1–27, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *ACM Symposium on Operating Systems Principles*, 2023.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-Embed: Improved techniques for training LLMs as generalist embedding models. In *International Conference on Learning Representations*, 2025.

Huaxia Li, Haoyun Gao, Chengzhang Wu, and Miklos A. Vasarhelyi. Extracting financial data from unstructured sources: Leveraging large language models. *Journal of Information Systems*, 39(1): 135–156, 2025a.

Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. TableBank: Table benchmark for image-based table detection and recognition. In *Twelfth Language Resources and Evaluation Conference (LREC)*, 2020.

Xuan Li, Jialiang Dong, and Raymond Wong. From surface to semantics: Semantic structure parsing for table-centric document analysis. In *Frontiers in Artificial Intelligence and Applications*, pp. 3234–3241, 2025b. doi: 10.3233/FAIA251190.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, 2024.

Weihong Lin, Zheng Sun, Chixiang Ma, Mingze Li, Jiawei Wang, Lei Sun, and Qiang Huo. Tsrformer: Table structure recognition with transformers. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, pp. 6473–6482, 2022. doi: 10.1145/3503161.3548038. URL https://doi.org/10.1145/3503161.3548038.

Ahmed Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter Staar. TableFormer: Table structure understanding with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4614–4623, 2022.

OpenAI. Update to GPT-5 system card: GPT-5.2, December 2025. URL https://cdn.openai.com/pdf/3a4153c8-c748-4b71-8e31-aecbde944f8d/oai_5_2_system-card.pdf.

ShengYun Peng, Aishwarya Chakravarthy, Seongmin Lee, Xiaojing Wang, Rajarajeswari Balasubramaniyan, and Duen Horng Chau. Unitable: Towards a unified framework for table recognition via self-supervised pretraining. *arXiv preprint arXiv:2403.04822*, 2024.

Marek Polewczyk and Marco Spinaci. ClusterTabNet: Supervised clustering method for table detection and table structure recognition. In *International Conference on Document Analysis and Recognition*, pp. 334–349. Springer, 2024.

Qing Shi, Jing He, Qiaosheng Chen, and Gong Cheng. DSEBench: A test collection for explainable dataset search with examples. *arXiv preprint arXiv:2510.17228*, 2025. URL https://arxiv.org/abs/2510.17228.

Brandon Smock, Rohith Pesala, and Robin Abraham. PubTables-1M: Towards comprehensive table extraction from unstructured documents. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Ting Sun, Cheng Cui, Yuning Du, and Yi Liu. PP-DocLayout: A unified document layout detection model to accelerate large-scale data construction. *arXiv preprint arXiv:2503.17213*, 2025. URL https://arxiv.org/abs/2503.17213.

Zineng Tang, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal. Unifying vision, text, and layout for universal document processing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Haochen Li, Jiale Zhu, Jiali Chen, Jiaxing Xu, Jiazheng Xu, Jing Chen, Jinghao Lin, Jinhao Chen, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Ruiliang Lyu, Shangqin Tu, Sheng Yang, Shengbiao Meng, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wei Jia, Wenkai Li, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyu Zhang, Xinyue Fan, Xuancheng Huang, Yadong Xue, Yanfeng Wang, Yanling Wang, Yanzi Wang, Yifan An, Yifan Du, Yiheng Huang, Yilin Niu, Yiming Shi, Yu Wang, Yuan Wang, Yuanchang Yue, Yuchen Li, Yusen Liu, Yutao Zhang, Yuting Wang, Yuxuan Zhang, Zhao Xue, Zhengxiao Du, Zhenyu Hou, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2026. URL https://arxiv.org/abs/2507.01006.

Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, Bo Zhang, Liqun Wei, Zhihao Sui, Wei Li, Botian Shi, Yu Qiao, Dahua Lin, and Conghui He. Mineru: An open-source solution for precise document content extraction, 2024a. URL https://arxiv.org/abs/2409.18839.

Dongsheng Wang, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu. DocLLM: A layout-aware generative language model for multimodal document understanding. In *Annual Meeting of the Association for Computational Linguistics*, 2024b.

Tingting Wang, Shixun Huang, Zhifeng Bao, J. Shane Culpepper, Shazia Sadiq, Volkan Dedeoglu, and Reza Arablouei. Unified data discovery across query modalities and user intents. *arXiv preprint arXiv:2604.27252*, 2026. URL https://arxiv.org/abs/2604.27252.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13484–13508, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.754. URL https://aclanthology.org/2023.acl-long.754/.

Ian Wu, Sravan Jayanthi, Vijay Viswanathan, Simon Rosenberg, Sina Khoshfetrat Pakazad, Tongshuang Wu, and Graham Neubig. Synthetic multimodal question generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 12960–12993, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.759. URL https://aclanthology.org/2024.findings-emnlp.759/.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Tongliang Li, Zhoujun Li, and Guanglin Niu. TableBench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25497–25506, 2025. doi: 10.1609/aaai.v39i24.34739. URL https://doi.org/10.1609/aaai.v39i24.34739.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CfXh93NDgH.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. LayoutLM: Pre-training of text and layout for document image understanding. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. Modeling localness for self-attention networks. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. In *European Conference on Computer Systems*, 2025.

Jiayi Yuan, Hongyi Liu, Shaochen Zhong, Yu-Neng Chuang, Songchen Li, Guanchu Wang, Duy Le, Hongye Jin, Vipin Chaudhary, Zhaozhuo Xu, Zirui Liu, and Xia Hu. KV cache compression, but what must we give in return? a comprehensive benchmark of long context capable approaches. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4623–4648, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.266. URL https://aclanthology.org/2024.findings-emnlp.266/.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6588–6600. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.acl-long.454.

Xinyi Zheng, Douglas Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. Global table extractor (GTE): A framework for joint table identification and cell structure recognition using visual context. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: Data, model, and evaluation. In *European Conference on Computer Vision*, pp. 564–580, 2020. doi: 10.1007/978-3-030-58589-1_34.

A SUPPLEMENTARY MATERIAL

This appendix is organized into four blocks. Appendix A.1 collects dataset-construction details, including the DEGCONTAB source statistics, source construction, split protocol, document-synthesis prompt, condition-generation prompt, and the machine-assisted generation pipeline with its human checking protocol. Appendix A.2 reports the Qwen and PCRC inference prompts. Appendix A.3 fixes the scoring set, label version, and metric conventions. Appendix A.4 gives the implementation details, the judgment-only view behind the judgment-F_1 figures of the main text, the context-selection and early-stopping ablations, the generic-cache and timing comparison, backbone generality, an extraction-only context ablation, and a closing discussion.

A.1 DATASET CONSTRUCTION DETAILS

A.1.1 EXAMPLE SCHEMA

Table 5 summarizes the common example structure used by FINCONTAB, PUBCONTAB, and DEGCONTAB.

Table 5: Common example schema for the three conditioned table extraction datasets.

Field	Description
Document	One source document, represented as an ordered sequence of page images.
Pages	Each page image contains zero or more tables and surrounding paragraph text.
Tables	Each table has a table identifier, crop, OCR text, and structured HTML annotation.
Paragraphs	Surrounding paragraph text is retained as textual annotation for document context.
Conditions	Each document is paired with three natural-language conditions.
Labels	For every condition, every table receives a binary `yes` or `no` label.
Target output	A system returns HTML only for tables labelled `yes` under the given condition.

A.1.2 SOURCE CONSTRUCTION

FINCONTAB, PUBCONTAB, and DEGCONTAB follow source-specific construction paths while sharing the same condition layer and checking protocol. FINCONTAB is derived from FINTABNET, a widely used benchmark for financial table extraction and table structure recognition (Zheng et al., 2021; Nassar et al., 2022; Huang et al., 2023; Peng et al., 2024; Polewczyk & Spinaci, 2024). We directly reuse the source documents and table annotations, and add the condition layer described in Appendices A.1.5–A.1.7. PUBCONTAB is derived from PubTables-style scientific table sources whose PubTabNet/PubTables-1M lineage is widely used for table extraction systems and metrics (Zhong et al., 2020; Smock et al., 2022; Nassar et al., 2022; Lin et al., 2022; Peng et al., 2024; Polewczyk & Spinaci, 2024). We likewise reuse the source documents and table annotations directly, and add the same condition layer. For DEGCONTAB, document-level synthesis and controlled challenge attributes are detailed in Appendix A.1.3.

Annotation quality is checked with the protocol of Appendix A.1.6: five unpaid volunteer annotators check a stratified sample of conditions against the reference HTML, a pre-selected fifth of the sample is labelled independently by a second annotator, and disagreements are settled by a third. Labels that fail the check are corrected before release, and conditions that cannot be decided are excluded from scoring.

A.1.3 DEGCONTAB SOURCE CONSTRUCTION

The source behind DEGCONTAB is generated at the document level. For each document, we first define a topic and collect table-structure templates with controlled header hierarchy, ruling-line and border style, and visual degradation settings. Gemini 3.5 Flash then generates a multi-page document for that topic, with its HTML tables embedded in the running text of the document. The renderer lays the document out as PDF pages with a machine-readable text layer, rasterizes them into page images, and applies the requested visual degradation, such as watermarks, stains, and stamps. Sampled documents are manually checked for coherence among the topic, paragraphs, table content, and rendered visual attributes. For each document, we retain the ordered page images,

Table 6: Scale and controlled challenge attributes of the source used for DEGCONTAB. Documents are generated directly as multi-page topic-level documents.

Aspect	Values	Share/size
Scale	Documents / rendered pages / table instances	23K / 230K / 386K
Header hierarchy	L1 / L2 / L3 / L4 / L5	25 / 35 / 20 / 12 / 8%
Visual degradation	None / watermark / stain / stamp	70 / 10 / 10 / 10%
Border style	Full / three-line / header-only / other	30 / 25 / 15 / 30%

table crops, source HTML, OCR text, and surrounding paragraphs; in the OCR text, characters under the injected watermarks, stains, and stamps are masked out, so degraded regions are observed only through the image.

This construction gives DEGCONTAB difficult table images while preserving reliable table-structure targets. In the main dataset construction, we then apply the same five condition strata used for FINCONTAB and PUBCONTAB; thus DEGCONTAB changes the visual and structural difficulty of the source tables without introducing a different condition distribution.

Table 6 summarizes the source scale and the controlled difficulty attributes. Header hierarchy Lx means that the table has x header levels, so larger values indicate deeper nested header structures. Visual degradation indicates whether the rendered table region is affected by page-level artifacts such as watermarks, stains, or stamps; the "none" group has no such injected artifact. Border style controls the ruling-line pattern: full borders enclose most cells, three-line tables use top/mid/bottom rules, header-only borders emphasize header separation, and "other" covers partial or non-standard ruling-line patterns.

Document splits. For FINCONTAB and PUBCONTAB, we keep the source document units and inherit the original train/dev/test split assignments from the underlying financial or scientific table-extraction sources. Thus the added condition layer does not change the source split protocol. For DEGCONTAB, the generated multi-page documents are split into train/dev/test at an 80/10/10 ratio, stratified by the controlled difficulty attributes in Table 6. In all three datasets, no document can appear in more than one split.

A.1.4 DEGCONTAB DOCUMENT SYNTHESIS PROMPT

The following template is used for DEGCONTAB document construction. The generator creates multi-page document content from a topic and a set of table-structure templates, while the renderer applies the requested ruling-line style and visual degradation.

```
[System]
You are generating source content for a controlled document table
benchmark.
[Input]
The target document topic is: {DOCUMENT_TOPIC}.
The document must use these table-structure templates:
{TABLE_STRUCTURE_TEMPLATES}.
The tables must follow these controlled difficulty settings:
    • header hierarchy level: {HEADER_LEVEL};
    • visual degradation type: {VISUAL_DEGRADATION};
    • ruling-line / border style: {BORDER_STYLE}.
[Goal]
Generate a realistic multi-page document on the target topic. The
document must contain several tables and surrounding paragraphs. Each
table should follow one of the supplied structure templates, match the
requested structural and visual settings, and be coherent with the
surrounding text.
[Instructions]
    1. Generate a document title and multiple ordered pages.
```

```
    2. Place one or more HTML tables on each page when appropriate.
    3. Write the surrounding paragraphs according to the natural
       relationship between text and tables in real documents of this
       kind; there is no need to deliberately avoid table-related
       content, nor to deliberately describe the tables.
    4. Make each table structure match its assigned template and the
       requested header hierarchy level.
    5. Make each table compatible with the requested ruling-line /
       border style, but do not write visual-degradation labels into
       table content.
    6. Treat visual degradation as rendering metadata; it will be
       applied by the renderer.
    7. Return only valid JSON.

[Output Schema]

{
  "document_title": "...",
  "pages": [
    {
      "page_index": 1,
      "paragraphs": ["...", "..."],
      "tables": [
        {
          "template_id": "...",
          "table_html": "<table>...</table>"
        }
      ]
    }
  ],
  "render_controls": {
    "header_level": "{HEADER_LEVEL}",
    "degradation": "{VISUAL_DEGRADATION}",
    "border": "{BORDER_STYLE}"
  }
}
```

A.1.5 CONDITION CONSTRUCTION STRATA

Table 7 gives the five balanced condition categories used for FINCONTAB, PUBCONTAB, and DEGCONTAB. The target shares specify the intended balance of the constructed datasets rather than a claim about naturally occurring query frequencies.

A.1.6 CONDITION GENERATION AND VALIDATION

Why conditions are generated from reference HTML only. The generator sees the official table HTML of a document, never the surrounding paragraphs. This is a deliberate and conservative choice. Had the generator also been given the paragraphs, it could have written conditions that are answerable *only* from those paragraphs, which would disadvantage table-only methods by construction and would make any measured benefit of context an artifact of the benchmark. Under the present design, the information needed for a label is in principle present in the reference HTML, so any gain from context must arise during inference rather than being built into the labels.

This design does not make the task trivial for a table-only system. The reference HTML is used for condition generation, label checking, and scoring only; apart from the one fixed formatting demonstration, evaluated models never receive reference HTML. They operate on rendered table images and unstructured OCR, from which row and column scope, nested header dependencies, units, and visually degraded cells are often hard to recover reliably. Our claim is accordingly not that every instance is unanswerable from the table alone, but that condition–table relevance is difficult to determine reliably from an isolated table, and that document context supplies useful auxiliary evidence.

Table 7: Condition construction strata. Target shares balance the constructed datasets across ability types rather than describe natural query frequencies.

#	Category	Target share	Condition focus	Typical pattern
C1	Lookup & Filter	20%	Literal lookup or filtering over table entries, headers, or explicit attributes.	Direct lookup
C2	Numerical Reasoning	20%	Numerical operations such as aggregation, comparison, extrema, ordinal selection, or trend checking, where the central evidence is quantitative.	Numeric operation
C3	Multi-Constraint Conjunction	20%	Several independent constraints must hold simultaneously, without requiring an intermediate derived result.	Parallel constraints
C4	Multi-Step Reasoning	20%	Operator chains in which an intermediate result is needed before the final judgment.	Sequential steps
C5	Structural Conditions	20%	Conditions on table structure, such as hierarchy, row/column semantics, aggregation pattern, or organization form.	Structural grounding

Generation pipeline. Given the strata in Table 7, we construct conditions for all three datasets with an LLM-assisted, source-grounded pipeline, following synthetic instruction and QA data practice in which language models propose candidates and filtering plus human review converts them into usable data (Wang et al., 2023; Honovich et al., 2023; Xu et al., 2024; Wu et al., 2024). For each reconstructed document, we provide the official HTML annotations of all tables to the generator. This avoids introducing OCR or visual-recognition noise during condition generation, while keeping table content and structure explicit in a compact textual form. Each generation call specifies one target stratum from Table 7; for each dataset, calls are scheduled so that the condition pool follows the target category distribution.

The generator returns a candidate condition and a binary label for every table in the document, so a condition may match zero, one, or multiple tables. We use the LLM output as a proposal rather than as ground truth. Candidate condition-label sets must be grounded in the provided table HTML, deterministic under a yes/no interpretation, and free of external knowledge requirements. We then apply automatic checks for malformed outputs, duplicates, ambiguous or under-specified wording, and label inconsistencies to every condition, followed by human checking of a stratified sample against the table HTML and, where needed, the rendered document (below). Conditions and their initial table labels were generated with Gemini 3.5 Flash. In total, approximately 1.29 million conditions were constructed from 429K documents. The final datasets keep the document images, document identifier, table identifiers, official table HTML annotations, condition text, condition category, and the positive/negative table labels for each condition. Splits are made at the document level so that conditions and tables from the same source document cannot appear in different splits.

Human checking protocol. A stratified 5% sample of conditions, 64,385 in total (689 on FINCONTAB, 60,247 on PUBCONTAB, 3,449 on DEGCONTAB), is drawn within dataset $\times$ category strata with a fixed seed and checked by human annotators. The unit of sampling is the condition, and for every sampled condition the annotator examines the labels of *all* tables of its document, so the unit of a checked label is a (document, condition, source table) triple; the number of sampled conditions and the number of checked labels are therefore reported separately (Table 8): the 64,385 conditions cover 217,229 condition–table pairs, in proportion to the tables per document of each dataset (24.7, 2.4, and 16.8 on average over the full FINCONTAB, PUBCONTAB, and DEGCONTAB). The original

condition text, identifiers, and automatic labels are frozen as L_0; human decisions are recorded in separate fields and never overwrite L_0.

Five annotators took part. Each sampled condition was assigned to one primary reviewer, so each annotator was the primary reviewer of about 12,900 conditions. A pre-selected 20% of the sample (12,877 conditions, drawn within the same strata, covering 43,449 pairs) was additionally labelled by a second reviewer working independently, and disagreements were settled by a third annotator who had not labelled the item. Reviewers saw only the condition, the reference HTML of the document's tables, and, when needed to locate a table or its content, the rendered pages; they did not see the automatic labels, the generator's rationale, the other reviewer's decision, or the output of any evaluated method. The annotators acted solely as technical validators; the datasets contain no human-subject records or sensitive personal data.

All reviewers applied the definition used throughout: a condition–table pair is `yes` when the condition is clearly satisfied by the reference HTML and `no` when it is clearly not. A multi-constraint condition requires every constraint to hold; numeric comparisons respect units, header scope, and the range of any aggregation; blank cells are not read as zero; when the rendered table is unreadable but the reference HTML is determinate, the HTML decides. A condition that requires a field the table demonstrably lacks is `no`. A pair whose condition is ambiguous, lacks the reference information needed to decide, or conflicts with the source annotation is marked U (undecidable) with a stated reason, never defaulted to `no`.

The final human label H is the primary reviewer's label on singly reviewed conditions, the shared label where two reviewers agree, and the adjudicator's label otherwise; the pre-adjudication labels are retained. When an automatic label is wrong, only the label is corrected; the condition text is never rewritten. A condition that remains undecidable after adjudication is excluded from the scoring of every method as a whole; when only the reference annotation of one table is defective, only the affected condition–table pairs are excluded. Exclusion counts and reasons are recorded (Table 8); valid hard cases are never removed. For each checked label we keep the dataset, document, condition, and source-table identifiers, L_0, the identifiers and labels of both reviewers, the adjudicator and H, the location of the deciding evidence (cell or page), and, for errors and disagreements, a short note. The remaining 95% of the conditions are automatically generated and automatically checked; they are not claimed to be human-confirmed item by item.

Agreement statistics. Table 8 reports, per dataset, the cross-tabulation of the automatic label L_0 against the final human label H over checked pairs with a binary H: a (automatic `yes`, human `yes`), b (`yes`, `no`), c (`no`, `yes`), and d (`no`, `no`). Three rates follow: the automatic–human agreement $(a+d)/(a+b+c+d)$; the positive mislabel rate $b/(a+b)$, the share of automatic positives that the reviewers rejected; and the negative miss rate $c/(c+d)$, the share of automatic negatives that the reviewers accepted. (The false-negative rate with respect to the human positives, $c/(a+c)$, is a different quantity and is not used.) Inter-annotator agreement is computed on the doubly labelled subset only, from the pre-adjudication labels: the denominator counts pairs to which both reviewers gave a binary label and the numerator those on which they agree; pairs on which either reviewer gave U are listed separately and are not counted as agreement. Overall values pool the counts across datasets rather than averaging per-annotator percentages. All figures describe the checked sample and are not extrapolated to error totals over the full datasets.

On this sample, the automatic labels and the final human labels agree on 98.5% of the 215,394 pairs with a binary H (99.2% on FINCONTAB, 98.2% on PUBCONTAB, and 99.2% on DEGCONTAB). Errors are rare in both directions: the reviewers rejected 2.4% of the automatic positives (2.2% / 2.4% / 3.0%) and accepted 0.9% of the automatic negatives (0.8% / 1.2% / 0.6%), that is, 1,821 labels were corrected from `yes` to `no` and 1,324 from `no` to `yes`. The lower agreement on PUBCONTAB reflects its balanced labels: with half of the pairs positive, the positive class, on which the automatic labels err more often, carries more weight. The two independent reviewers agreed on 98.5% of the 43,189 doubly labelled pairs with two binary labels (98.8% / 98.2% / 99.1%), and one of them marked U on 260 further pairs (0.6%). After adjudication, 462 conditions (0.7% of the sample) remained undecidable and were excluded as a whole, and 302 pairs were excluded for a defective reference annotation. For all checked pairs, H replaces L_0 in scoring (Appendix A.3).

Table 8: Human checking of the condition labels on the stratified 5% sample: counts of checked (document, condition, source table) pairs by automatic label L_0 (rows `yes`/`no`) and final human label H, the rates derived from them, the doubly labelled subset, and the exclusions. Overall values pool the counts.

	FINCONTAB	PUBCONTAB	DEGCONTAB	Overall
Sampled conditions	689	60,247	3,449	64,385
Checked pairs (all)	17,009	141,782	58,438	217,229
Checked pairs with binary H	16,851	140,527	58,016	215,394
a: L_0 `yes`, H `yes`	809	68,745	3,544	73,098
b: L_0 `yes`, H `no`	18	1,695	108	1,821
c: L_0 `no`, H `yes`	124	856	344	1,324
d: L_0 `no`, H `no`	15,900	69,231	54,020	139,151
Automatic–human agreement $(a+d)/(a+b+c+d)$	99.2%	98.2%	99.2%	98.5%
Positive mislabel rate $b/(a+b)$	2.2%	2.4%	3.0%	2.4%
Negative miss rate $c/(c+d)$	0.8%	1.2%	0.6%	0.9%
Doubly labelled pairs, both binary	3,373	28,304	11,512	43,189
Inter-annotator agreement (pre-adjudication)	98.8%	98.2%	99.1%	98.5%
Doubly labelled pairs with a U	25	171	64	260
Conditions excluded (undecidable)	4	433	25	462
Pairs excluded (defective reference)	59	239	4	302

A.1.7 CONDITION GENERATION PROMPT

The following template is used for source-grounded condition construction. For each document, `{DOCUMENT_TABLES}` is filled with the reference HTML of all tables in page order. For each generation call, `{TARGET_CONDITION_TYPE}` is filled with one sampled condition type from Table 7. The generator is not given surrounding paragraphs or document images.

```
[System]
You are constructing data for conditioned table extraction.
[Input]
The document contains multiple tables.  For each table, you are given
only its reference HTML. The tables are listed in document page order.
The target condition type for this call is:  {TARGET_CONDITION_TYPE}.
[Goal]
Generate one natural-language condition of the target type for
conditioned table extraction in this document.  Assign a yes/no label
to every table.  The condition may match zero, one, or multiple
tables.
[Instructions]

    1. Generate exactly one condition for the document.
    2. The condition must follow the supplied target condition type.
    3. Each condition must be answerable from the provided table HTML
       strings.
    4. Do not rely on external knowledge.
    5. Do not mention table IDs, page numbers, file names, or
       annotation fields.
    6. Avoid trivial conditions that match all tables unless the
       evidence makes such a condition meaningful.
    7. Avoid conditions that are ambiguous, underspecified, or
       impossible to judge from the provided tables.
    8. For each condition, label every table as "yes" or "no".
    9. Return only valid JSON.

[Output Schema]
```

```
{
  "condition": "...",
  "category": "{TARGET_CONDITION_TYPE}",
  "table_labels": [
    {"table_id": "T1",
     "label": "yes|no",
     "evidence": "..."},
    {"table_id": "T2",
     "label": "yes|no",
     "evidence": "..."}
  ]
}
```

[Document Tables]
{DOCUMENT_TABLES}

A.2 Inference Prompts

We use the following prompt templates for the Qwen-based single-shot baseline and for PCRC. The placeholders are filled with the corresponding table crop, OCR text, condition, related paragraphs, and one-shot formatting example. In the PCRC template the blocks up to the shared related paragraphs form the fixed head of Section 3.2; the table image, table OCR, and exclusive related paragraphs form the table-specific tail, and the extraction turn, carrying the sample table image and sample HTML, is appended only after a positive judgment.

A.2.1 Single-Shot Qwen Prompt

[System]
You are performing conditioned table extraction.
[Input]
You are given:

- a table image;
- OCR text from this table image. The OCR text contains table characters only and is not a structured table;
- a natural-language condition;
- one sample table image;
- the reference HTML for the sample table.

[Goal]
First decide whether the table satisfies the condition. If it does, extract the table as HTML by following the correspondence between the sample table image and its reference HTML. If it does not, return no.
[Instructions]

1. Use the table image to recover layout, row/column structure, headers, and cell spans.
2. Use the OCR text only as unstructured textual evidence for the table content.
3. Use the sample table image and sample HTML only as a formatting demonstration; do not copy their content.
4. If the candidate table satisfies the condition, return only the extracted HTML table.
5. If the candidate table does not satisfy the condition, return exactly no.

[Table Image]
{TABLE_IMAGE}
[Table OCR]
{TABLE_OCR}
[Condition]

{CONDITION}
[Sample Table Image]
{SAMPLE_TABLE_IMAGE}
[Sample HTML]
{SAMPLE_HTML}

A.2.2 PCRC PROMPT

[System]
You are a context-aware conditioned table extractor.
[Input]
You are given:

- a natural-language condition;
- paragraphs shared across the tables of this document;
- a table image;
- table OCR, containing characters only and no structure;
- paragraphs specific to this table;
- once the table is accepted, a sample table image and its reference HTML.

[Goal]
Use the image, OCR, and related paragraphs to decide whether the table satisfies the condition: answer yes if it clearly does, no if it clearly does not, and uncertain if the evidence is insufficient. An accepted table is then extracted as HTML following the sample image/HTML correspondence.
[Instructions]

1. Use the table image to recover visual layout, row/column structure, headers, and cell spans.
2. Use the OCR text as unstructured textual evidence for table content.
3. Use related paragraphs for table topic, variable meanings, measurement context, and header scope.
4. For the judgment, return exactly yes, no, or uncertain.
5. For extraction, use the sample table image and sample HTML only as a formatting demonstration; do not copy their content. Return only the extracted HTML table.

[Condition]
{CONDITION}
[Shared Related Paragraphs]
{SHARED_PARAGRAPHS}
[Table Image]
{TABLE_IMAGE}
[Table OCR]
{TABLE_OCR}
[Exclusive Related Paragraphs]
{EXCLUSIVE_PARAGRAPHS}
[Extraction Turn, appended after a positive judgment]
[Sample Table Image]
{SAMPLE_TABLE_IMAGE}
[Sample HTML]
{SAMPLE_HTML}
[Extraction Request]
Extract the table as HTML following the sample image/HTML correspondence.

A.3 Evaluation Protocol

This section fixes the scoring set, the label version, and the metric conventions shared by every method in Section 5 and in the appendix.

Scope. We evaluate condition judgment and table reconstruction on a fixed set of upstream table candidates that can be uniquely aligned to the source annotations. All methods use the same candidate-to-source mapping, reference HTML, label version, and scoring set. Unmatched candidates are excluded from quality scoring, and source tables without matched candidates are outside the recall denominator. The reported quality therefore measures performance conditional on the source-aligned candidate set rather than end-to-end table detection and extraction. For each retained condition, the cost measurements include all computation incurred when processing the original candidate set, including candidates excluded from quality scoring.

Frozen inputs and identifiers. The upstream candidate set $\mathcal{C}_{\mathrm{raw}}$ of every document, its crops, OCR, and paragraph blocks, and all model outputs are frozen under stable candidate identifiers; every source table carries a source-table identifier, document and page identifiers, a reference box, and its reference HTML. Reference HTML is used only for human checking and scoring; apart from the one fixed formatting demonstration, it is never part of a model input. The alignment below serves scoring only: it neither re-crops images nor alters the context that any run consumed.

Candidate-to-source alignment. Where a traceable and verified mapping from candidates to source-table identifiers exists, it is used directly. Only candidates without such a mapping are aligned by box matching within the same document and page. Candidate and source boxes are first brought to a common rotation, origin, and scale (for DegConTab the source boxes follow the same image transformation as the rendered page); candidate–source pairs with IoU ≥ 0.5 are admissible, and a one-to-one matching over admissible pairs is chosen that first maximizes the number of matched pairs and then the total IoU, with exact ties broken by stable identifiers. The threshold is fixed at 0.5 and not tuned. Tables that span pages keep the table division and HTML of the source annotation; no predicted tables are stitched.

Scoring set. The result is the scorable set $\mathcal{C}_{\mathrm{eval}}$, in which each source table corresponds to at most one candidate and each candidate to at most one source table. The matching is computed once, without reference to condition labels, method predictions, or extraction scores, and is shared by all conditions and all methods. Table 9 lists the fixed handling of every alignment case. Unmatched candidates enter neither the numerator nor the denominator of any quality score and are not counted as negatives.

Table 9: Alignment cases between upstream candidates and source tables and their fixed handling in the scoring set $\mathcal{C}_{\mathrm{eval}}$.

Case	Handling
One candidate uniquely aligned to one source table	The candidate inherits the condition labels and the full reference HTML of that source table and enters $\mathcal{C}_{\mathrm{eval}}$.
Several duplicate candidates on one source table	The one-to-one matching admits exactly one; the others are recorded as unmatched candidates, and the source table is scored once.
One source table split into several candidates	At most one candidate is admitted under the same matching rule; predictions are not concatenated, scoring uses the full source HTML, and no candidate is admitted below the threshold.
One candidate covering several source tables	At most one source table is assigned under the same rule; its prediction is not copied to the other source tables and earns no additional true positive.
Candidate without a source table, or reference HTML unavailable	Recorded with its reason and excluded from quality scoring; it is not labelled `no`.
Source table without a matched candidate (including missed detections)	Outside the recall denominator of this fixed-candidate protocol; reported separately as an unmatched source table.

Labels. Every (condition, candidate) pair in $\mathcal{C}_{\text{eval}}$ takes the label of its source table under that condition: the final human label H where the pair was checked (Appendix A.1.6) and the frozen automatic label otherwise. All methods use the same label version and the same exclusion list of undecidable conditions and defective pairs.

Prediction conventions. The output of each pipeline is mapped to a selection decision as defined in Section 5.1: an explicit rejection is `no`, an acceptance is `yes`, and an accepted table whose HTML later proves defective keeps its `yes`. A run that was executed but returned no valid decision is a task failure: the table counts as not selected, so a positive becomes a false negative, and failures are counted separately; a missing run record is never treated as a `no` prediction.

Metrics. Over all valid pairs in $\mathcal{C}_{\text{eval}}$ we accumulate TP, FP, FN, and TN per dataset and report micro $P = \text{TP}/(\text{TP}+\text{FP})$, $R = \text{TP}/(\text{TP}+\text{FN})$, and $F_1 = 2\,\text{TP}/(2\,\text{TP}+\text{FP}+\text{FN})$. Extraction is scored on true positives only, with the unmodified TEDS implementation and one HTML normalization for every method; model outputs are never repaired against the reference. An accepted table whose extraction failed, is empty, or cannot be parsed scores TEDS 0 and stays in the denominator. With n_j the number of reference cells of true positive j, $\text{EQ} = \sum_j n_j\,\text{TEDS}_j / \sum_j n_j$; the composite scores are $P_c = P \cdot \text{EQ}$, $R_c = R \cdot \text{EQ}$, and their harmonic mean, which equals $F_1 \cdot \text{EQ}$. When there is no true positive, EQ and the composite scores are 0; every other empty denominator also yields 0.

Cost. Cost is aggregated over the same retained conditions. For each retained condition, every call made while processing its original candidate set is counted, including calls on candidates outside $\mathcal{C}_{\text{eval}}$; a condition excluded as undecidable is removed from both the quality and the cost aggregates of every method.

Processing counts. Table 10 reports, per dataset, the number of source tables, upstream candidates, scorable matches, unmatched candidates, unmatched source tables, exclusions, and task failures. The table counts in Table 2 are source tables; scoring uses the matched candidates. The scorable set covers 89.8%, 96.4%, and 93.3% of the source tables of FINCONTAB, PUBCONTAB, and DEGCONTAB (101,325, 913,235, and 360,196 matched tables); the remaining 11,562, 34,407, and 25,804 source tables are assigned no candidate by the one-to-one matching. On the candidate side, 27.5%, 8.7%, and 11.0% of the upstream candidates are unmatched. The larger share on FINCONTAB comes from financial pages, where the detector also boxes table-like text blocks next to the tables; such candidates are processed by every method and counted in its cost, but do not enter the quality scores. Exclusions are confined to the checked sample: 462 conditions and 302 pairs in total (Table 8). Task failures are negligible: 38, 204, and 146 pair-level runs summed over the nine methods of Table 3, about one run in 10^5 on every dataset.

Table 10: Processing counts of the evaluation protocol per dataset. Scorable matches is $|\mathcal{C}_{\text{eval}}|$; exclusions follow Table 8; task failures are pair-level runs without a valid decision, summed over the nine methods of Table 3.

	FINCONTAB	PUBCONTAB	DEGCONTAB
Source tables	112,887	947,642	386,000
Upstream candidates $\lvert\mathcal{C}_{\text{raw}}\rvert$	139,681	1,000,308	404,847
Scorable matches $\lvert\mathcal{C}_{\text{eval}}\rvert$	101,325	913,235	360,196
Unmatched candidates	38,356	87,073	44,651
Unmatched source tables	11,562	34,407	25,804
Conditions excluded (undecidable)	4	433	25
Pairs excluded (defective reference)	59	239	4
Task failures (summed over methods)	38	204	146

A.4 SUPPLEMENTARY EXPERIMENTS

This subsection reports auxiliary experimental views that support the main composite-quality and compute results in Section 5.

Table 11: Judgment-only quality. P, R, and F_1 are micro precision, recall, and F_1 on the positive class. FIN, PUB, and DEG denote FINCONTAB, PUBCONTAB, and DEGCONTAB, respectively. Best per dataset per column in **bold**.

	FIN			PUB			DEG		
Method	P	R	F_1	P	R	F_1	P	R	F_1
TATR+TAPAS	0.045	0.726	0.086	0.656	0.833	0.734	0.048	0.746	0.090
TATR+Qwen	0.612	0.605	0.609	0.679	0.852	0.756	0.612	0.589	0.600
MinerU+Qwen	0.541	0.704	0.612	0.540	0.743	0.626	0.575	0.703	0.633
MinerU+TAPAS	0.061	0.520	0.110	0.430	0.620	0.508	0.043	0.656	0.081
Qwen	0.666	0.802	0.727	0.642	0.844	0.730	0.697	0.766	0.730
PCRC (ours)	**0.888**	**0.914**	**0.901**	**0.884**	**0.919**	**0.901**	**0.899**	**0.906**	**0.902**
PCRC w/o cache	**0.888**	**0.914**	**0.901**	**0.884**	**0.919**	**0.901**	**0.899**	**0.906**	**0.902**

A.4.1 IMPLEMENTATION DETAILS

All experiments reported in the main text and appendix use the following implementation and hardware configuration. All Qwen-based modules share the same Qwen2.5-VL-7B-Instruct checkpoint with greedy decoding and the same generation budget. TATR and MinerU are run with their official released weights and default configurations.

Relevance encoder. The embedding encoder ϕ of Equation 1 is the pretrained NV-Embed model (Lee et al., 2025), applied to the OCR text of paragraphs and tables without task-specific fine-tuning.

Upstream parsing. For all three datasets, candidate table regions and paragraph blocks are produced by PP-Structure from the PaddleOCR toolkit (Cui et al., 2025a; Sun et al., 2025). FINCONTAB and DEGCONTAB derive from source PDFs that carry a machine-readable text layer, so the text inside those regions is read with PyMuPDF instead of running optical recognition; for DEGCONTAB, text under the rendered watermarks, stains, and stamps is masked out, so degraded regions are observed only through the image.

Selection budgets and thresholds. Unless a sweep states otherwise, all reported runs use $K_{\text{shared}} = 4$, $\tau_{\text{shared}} = 0.12$, $K_{\text{excl}} = 3$, and $\tau_{\text{excl}} = 0.3$. The sensitivity of these four values is reported in Table 15.

Condition generation. Candidate conditions and their initial labels are proposed by Gemini 3.5 Flash, filtered automatically, and checked by human annotators on a stratified sample, as described in Appendix A.1.6.

Each evaluation server is configured as PyTorch 2.8.0 with CUDA 12.8 on Python 3.12 (Ubuntu 22.04); 8 × NVIDIA RTX PRO 6000 (96 GB) per node; 176 vCPU Intel Xeon Platinum 8470Q with 880 GB system memory. The generic prefix-cache comparison and the timing of Table 16 run on vLLM 0.27.1 on one GPU of the same node.

A.4.2 JUDGMENT QUALITY

Table 11 reports judgment precision, recall, and F_1 on the positive class with micro aggregation over all scorable (condition, candidate) pairs (Appendix A.3); the best entry per dataset per column is bolded. To keep the table readable, the two pipelines that reuse the single-shot Qwen judge unchanged (Qwen+TATR and Qwen+MinerU) are omitted from this view: they share Qwen's judgments by construction and would only duplicate Qwen's row. PCRC w/o cache follows the same prompt content and progressive evidence schedule as PCRC, but recomputes the prefix instead of reusing cached states; it verifies that the proposed KV-cache reuse strategy leaves prediction quality unchanged.

Overall results. PCRC and PCRC w/o cache produce identical judgment quality: their precision, recall, and F_1 coincide on all three datasets. This confirms that the proposed KV-cache reuse strategy does not alter judgments; its benefit is computational and is quantified in Section 5.3. PCRC sits at the top of the judgment table: its judgment F_1 scores of 0.901, 0.901, and 0.902 on FINCONTAB,

PUBCONTAB, and DEGCONTAB are $+17.4$, $+17.1$, and $+17.2$ pp above single-shot Qwen, and at least $+14.5$ pp above the strongest non-PCRC modular pipeline on every dataset.

Qwen versus modular pipelines. The single-shot Qwen baseline provides the strongest general non-PCRC reference. Even without document context, direct VLM judging is usually stronger than modular extractor-mediated judging in both precision and recall. On FINCONTAB, Qwen reaches $P = 0.666$ and $R = 0.802$, while the best extractor-first pipeline, MinerU+Qwen, reaches only $P = 0.541$ and $R = 0.704$. On DEGCONTAB, Qwen reaches $P = 0.697$ and $R = 0.766$, again ahead of MinerU+Qwen's $P = 0.575$ and $R = 0.703$. The TAPAS-based pipelines perform particularly poorly, mainly because their precision is extremely low: on FINCONTAB, TATR+TAPAS has only $P = 0.045$ and MinerU+TAPAS has only $P = 0.061$. We attribute this pattern mainly to the extractor stage. TATR and MinerU are task-specialized non-VLM extractors whose structure predictions depend strongly on their pretraining distributions. When evaluated off those distributions, their extracted HTML often becomes noisy: a missing header, an incorrect span, or a dropped numeric token can turn the downstream judge's input into a distorted table. Direct VLM judging avoids this intermediate-structure bottleneck, reasons from the image itself, and benefits from the broader pretraining coverage and larger model capacity of a general-purpose VLM.

The only judgment-side departure from this pattern is PUBCONTAB: TATR+TAPAS (judgment $F_1 = 0.734$) and TATR+Qwen (judgment $F_1 = 0.756$) slightly exceed single-shot Qwen (judgment $F_1 = 0.730$). This is the expected favorable case for TATR rather than evidence against the VLM baseline. TATR was trained on PubTables-1M, the same source family covered by PUBCONTAB, so its extracted HTML is unusually reliable in this matched setting and can give the downstream judge a clean structured signal. The same TATR-based pipelines fall substantially in judgment F_1 on FINCONTAB and DEGCONTAB: TATR+TAPAS drops by -64.8 pp and -64.4 pp, while TATR+Qwen drops by -14.7 pp and -15.6 pp.

Qwen versus PCRC. PCRC uses the same Qwen backbone as the single-shot baseline, but augments table image/OCR evidence with table-relevant document context: shared related paragraphs are cached in the head, and exclusive related paragraphs are appended progressively when more evidence is useful. This raises judgment F_1 over Qwen by $+17.4$, $+17.1$, and $+17.2$ pp on FINCONTAB, PUBCONTAB, and DEGCONTAB. Related context acts as auxiliary evidence: it helps disambiguate a table's semantic scope, connect the table to surrounding titles and narrative statements, clarify variable meanings and measurement context, and separate visually similar candidate tables. This makes condition-table matching more stable than relying on the table crop alone.

Human-checked subset. Table 12 restricts scoring to the checked pairs of Table 8 that lie in the scorable set $\mathcal{C}_{\text{eval}}$, that is, to the pairs whose labels were verified by humans (their labels are already H in the main protocol), using each method's frozen predictions; no model is re-run. Because the restriction changes the set of true positives, EQ is re-aggregated over the true positives of the subset from the saved HTML with the same TEDS evaluator, and the three methods, Qwen, PCRC, and the single-judgment protocol of Appendix A.4.5, are compared on exactly the same pairs. On this fully human-verified subset PCRC leads single-shot Qwen by 17.1–17.8 judgment-F_1 points and 12.0–15.6 composite points, and PCRC and the single-judgment protocol remain within 1.8 judgment points of each other (single judgment minus PCRC: -0.1, $+1.8$, and $+0.1$ on FINCONTAB, PUBCONTAB, and DEGCONTAB), as in Table 14; the PCRC–Qwen ordering of Tables 3 and 11 is unchanged.

Table 12: Judgment F_1 / composite F_1-TEDS on the human-checked label subset: the checked pairs of Table 8 that lie in $\mathcal{C}_{\text{eval}}$ (labels H, as in the main protocol), scored from the frozen predictions of each method; the single-judgment protocol is that of Appendix A.4.5. Scored pairs / positives: 15,371 / 851 on FINCONTAB, 135,613 / 67,138 on PUBCONTAB, 53,915 / 3,610 on DEGCONTAB.

Method (F_1 / F_1-TEDS)	FINCONTAB	PUBCONTAB	DEGCONTAB
Qwen (single-shot)	0.735 / 0.605	0.729 / 0.601	0.731 / 0.645
PCRC	0.910 / 0.725	0.900 / 0.757	0.909 / 0.795
Single judgment + generic prefix cache	0.909 / 0.721	0.918 / 0.759	0.910 / 0.784

Table 13: Per-condition-category results (judgment F_1 / composite F_1-TEDS), computed within each stratum of Table 7 and averaged over the three datasets.

Method (F_1 / F_1-TEDS)	**C1** Lookup	**C2** Numeric	**C3** Multi-constr.	**C4** Multi-step	**C5** Structural
Qwen (single-shot)	0.716 / 0.598	0.752 / 0.627	0.754 / 0.650	0.750 / 0.625	0.685 / 0.580
PCRC	**0.912 / 0.765**	**0.913 / 0.778**	**0.913 / 0.785**	**0.883 / 0.744**	**0.886 / 0.748**

A.4.3 Where the Gains Come From

PCRC couples two mechanisms: it selects surrounding context as reference evidence, and it appends that evidence progressively, stopping as soon as the judgment is reliable. Ablating them separately shows that the two mechanisms buy different things, and that neither is redundant.

Quality comes from selection, not from adding text. Feeding the same number of paragraphs in random order lands at 0.71–0.73 judgment F_1, essentially the no-context baseline (≈ 0.73): PCRC's judge-then-extract prompt with uninformative evidence performs like single-shot Qwen, so the gain comes from what is selected rather than from how the prompt is organized. Loading *all* paragraphs reaches only 0.84–0.85 against full PCRC's ≈ 0.90: useful evidence is diluted by the unrelated majority of the page. Both halves of the relevance score matter, since semantic-only and spatial-only retrieval reach only 0.85–0.90, and the two evidence types are complementary, since shared-only (0.78–0.81) and exclusive-only (0.82–0.86) both fall below the full method.

Cost saving comes from early stopping. Removing the reliability gate, so that every exclusive paragraph is appended unconditionally, leaves quality essentially unchanged (0.898–0.916 against PCRC's 0.901–0.902, a change of -0.4 to $+1.5$ points): quality is attributable to selecting the right context rather than to how much of it is eventually loaded. Its cost, however, is higher: prefill rises from 33.0/13.4/19.6k to 40/17/24k, because evidence that early stopping would never have loaded is encoded for every table. Early stopping therefore buys compute at almost no cost in quality. Per-variant quality and cost numbers, the protocol of each variant, and a four-parameter sensitivity sweep follow in Appendix A.4.4.

The gains hold across all condition types. Table 13 breaks the results down over the five condition categories C1–C5 defined in Table 7. PCRC improves on the single-shot baseline in every category, by approximately 13–20 judgment F_1 points, with the largest gains on structural (C5) and lookup (C1) conditions. Multi-step reasoning (C4) and structural conditions (C5) remain the most challenging categories, with the lowest PCRC scores and the smallest gain on C4, consistent with their need for chained or hierarchy-aware inference rather than a single lookup.

The framework is not tied to one backbone. Porting the full PCRC contract to GLM-4.5V (V Team et al., 2026), a second open-weight backbone, raises judgment F_1 from 0.88–0.90 to 0.95–0.96 ($+5.4$ to $+7.7$ points) while cache reuse holds the recomputation ratio at 22–31%, reproducing the pattern observed on Qwen2.5-VL-7B. Results for GLM-4.5V and for three frontier VLMs evaluated as single-shot baselines are reported in Appendix A.4.7.

A.4.4 Ablation of Context Selection and Early Stopping

This section reports the per-variant numbers summarized in Appendix A.4.3. All variants use the Qwen backbone, and all retrieval variants except All context are budget-matched to the same maximum paragraph budget $K_{\text{shared}} + K_{\text{excl}}$, so that only the ranking or selection strategy differs. Table 14 reports the results; the variants are defined as follows.

- **Random context.** Paragraphs are drawn at random up to the budget and concatenated in random order with the table image, table OCR, and the rest of the PCRC prompt, all in a single round: the judge-then-extract protocol is kept, only the evidence is uninformative.
- **All context.** Every paragraph of the document is loaded into the head of the cache; for each table only the tail, holding the table image and OCR, is replaced.
- **Semantic-only / spatial-only retrieval.** Relevance uses only the semantic factor, or only the spatial factor, instead of their product.

Table 14: Ablation of context selection and early stopping (Qwen backbone). F_1 is judgment F_1; F_1-TEDS is the composite judgment-plus-extraction score. Pre. is per-condition prefill tokens and R.R. the recomputation ratio against its own no-cache run. All retrieval variants are budget-matched to the same maximum paragraph budget $K_{\text{shared}} + K_{\text{excl}}$; only the ranking or selection strategy changes. Random context is delivered in a single round with no cache reuse and therefore has no R.R.

	Quality (F_1 / F_1-TEDS)			**Cost** (Pre. (k) / R.R. (%))		
Variant	FIN	PUB	DEG	FIN	PUB	DEG
Random context	0.706 / 0.574	0.728 / 0.617	0.729 / 0.635	51.47 / —	28.63 / —	40.52 / —
All context	0.837 / 0.642	0.854 / 0.669	0.841 / 0.726	23.81 / 73.46	10.37 / 63.92	16.58 / 52.14
Semantic-only retrieval	0.861 / 0.693	0.868 / 0.721	0.895 / 0.754	34.76 / 41.29	14.83 / 31.57	23.64 / 30.91
Spatial-only retrieval	0.873 / 0.687	0.849 / 0.748	0.866 / 0.775	37.42 / 43.86	15.79 / 33.21	23.17 / 34.65
Shared-only context	0.805 / 0.672	0.783 / 0.699	0.809 / 0.727	17.98 / 53.71	8.94 / 66.23	15.48 / 52.69
Exclusive-only context	0.862 / 0.676	0.864 / 0.724	0.825 / 0.722	26.75 / 70.31	11.68 / 61.54	17.92 / 55.27
No reliability gating	0.907 / 0.731	0.916 / 0.759	0.898 / 0.786	40.19 / 27.16	17.45 / 26.96	23.96 / 23.09
Cache management						
PCRC schedule + generic prefix cache	0.901 / 0.723	0.901 / 0.756	0.902 / 0.794	33.38 / 28.66	13.47 / 29.13	20.11 / 25.01
Single judgment + generic prefix cache	0.902 / 0.722	0.918 / 0.759	0.905 / 0.784	42.68 / 67.89	18.82 / 70.92	26.43 / 66.31
PCRC (full)	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**

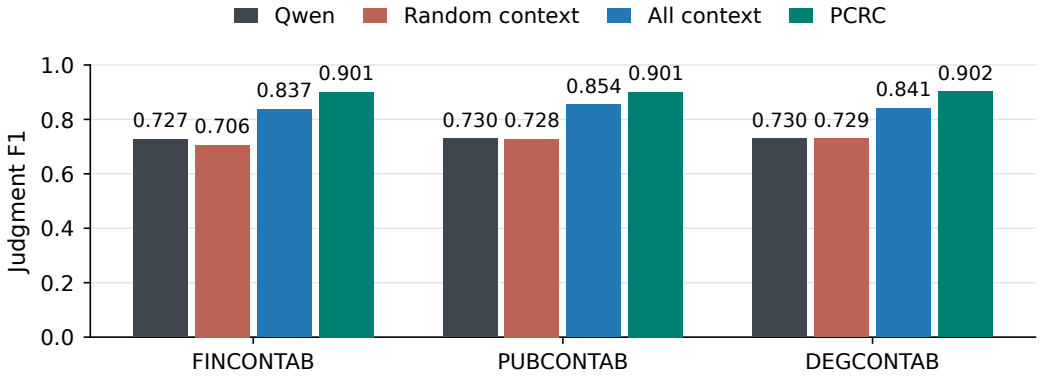


Figure 3: Effect of context selection on judgment quality. Random context does not improve over Qwen on these datasets. All context improves judgment F_1, but remains below PCRC, supporting the value of selecting relevant contextual evidence.

- **Shared-only / exclusive-only context.** Only shared paragraphs, or only table-specific exclusive paragraphs, are supplied as evidence.
- **No reliability gating.** All exclusive paragraphs are appended unconditionally, with no early stopping on a reliable judgment.

Figure 3 compares judgment F_1 for Qwen, random context, all context, and PCRC across the three datasets.

Hyper-parameter sensitivity. Table 15 varies each of the four selection parameters in turn while holding the others at their defaults. Quality moves smoothly across the tested ranges, and the worst cell in the entire sweep (judgment F_1 0.858, at the aggressive $\tau_{\text{excl}} = 0.6$) still far exceeds the single-shot baseline. The trends follow the expected direction, up to minor fluctuations on individual cells: larger budgets and looser thresholds admit more context and help slightly until saturation, after which extra context adds noise, while tighter thresholds save tokens but discard useful paragraphs. The defaults are not the best setting in the sweep: $K_{\text{excl}} = 2$ is slightly better on all three datasets, by 0.2–0.7 judgment F_1 and 0.1–1.2 composite points, with 5–14% fewer prefill tokens, and $K_{\text{excl}} = 4$

Table 15: Hyper-parameter sensitivity. The four blocks vary, in order, the shared-context budget K_{shared}, the exclusive-context budget K_{excl}, the shared threshold τ_{shared}, and the exclusive threshold τ_{excl}. Each block varies one parameter while keeping the other three at their defaults (bold rows, identical across blocks).

	Quality (F_1 / F_1-TEDS)			**Cost** (Pre. (k) / R.R. (%))		
Setting	FIN	PUB	DEG	FIN	PUB	DEG
Shared budget						
2	0.902 / 0.725	0.896 / 0.736	0.882 / 0.799	30.64 / 39.17	12.87 / 36.69	18.55 / 38.98
4 (default)	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**
6	0.909 / 0.731	0.903 / 0.760	0.910 / 0.799	38.14 / 28.39	17.43 / 29.92	23.24 / 23.79
8	0.905 / 0.721	0.902 / 0.757	0.885 / 0.778	43.76 / 24.52	17.61 / 25.56	28.75 / 22.42
Exclusive budget						
1	0.889 / 0.702	0.878 / 0.735	0.868 / 0.783	27.56 / 46.19	10.51 / 36.47	17.18 / 37.82
2	0.906 / 0.730	0.908 / 0.768	0.904 / 0.795	30.75 / 32.83	11.57 / 33.95	18.54 / 28.22
3 (default)	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**
4	0.903 / 0.739	0.910 / 0.760	0.911 / 0.807	46.11 / 25.49	20.94 / 25.19	26.94 / 22.63
Shared threshold						
0.06	0.903 / 0.725	0.909 / 0.762	0.906 / 0.801	37.81 / 29.21	15.34 / 27.12	22.91 / 26.26
0.12 (default)	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**
0.18	0.893 / 0.710	0.896 / 0.741	0.898 / 0.790	30.16 / 34.34	13.38 / 33.89	19.72 / 31.87
0.24	0.884 / 0.720	0.882 / 0.745	0.871 / 0.780	30.12 / 39.15	11.56 / 34.33	17.55 / 35.17
Exclusive threshold						
0.15	0.908 / 0.732	0.910 / 0.763	0.908 / 0.804	43.73 / 32.18	18.14 / 30.55	26.74 / 23.51
0.3 (default)	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**
0.45	0.867 / 0.714	0.890 / 0.730	0.863 / 0.776	26.47 / 39.22	12.32 / 37.13	16.23 / 39.39
0.6	0.866 / 0.690	0.880 / 0.710	0.858 / 0.760	24.34 / 41.86	10.58 / 40.55	15.73 / 40.28

or $\tau_{\text{excl}} = 0.15$ score marginally higher at roughly 30–60% more prefill. The defaults stay within one judgment-F_1 point and two composite points of the best setting in every column.

A.4.5 Comparison with Generic Prefix Caching

Serving engines reuse the KV states of prefixes that recur across requests through block-level prefix caching (Kwon et al., 2023). We compare PCRC's explicit head/tail management with this generic policy under two protocols, reported in the cache-management block of Table 14 and in Table 4. *PCRC schedule + generic prefix cache* replays PCRC's exact call sequence, with the same prompts, rounds, and outputs, through an engine whose automatic prefix cache decides what to keep, so quality is identical by construction and only the prefill actually computed changes. *Single judgment + generic prefix cache* supplies the shared and exclusive paragraphs selected by PCRC in one call, judges once, extracts the accepted tables, and relies on the generic cache to reuse the shared head across tables. Both run on vLLM 0.27.1 with prefix caching and multimodal block hashing enabled (block size 16); for every request we record the prompt tokens and the cached tokens, the prefill actually computed being their difference, and R.R. is taken against the same call sequence with caching disabled, as in Table 4.

With one document served at a time, the generic cache recomputes 25–29% of the no-cache prefill, within 0.7 points of PCRC's 24–29%: the reuse follows from the schedule, and PCRC's contract runs unchanged on a generic prefix cache. The single-judgment protocol reaches 0.902–0.918 judgment F_1, matching the variant without the reliability gate, and uses 30–40% more prefill than PCRC, which stops appending evidence once a judgment is reliable.

Measured inference time. Table 16 times the two protocols above on the same vLLM engine, so it compares progressive execution with a single judgment under one generic prefix cache, not the native head/tail implementation. Each dataset contributes 20 randomly drawn documents with one pre-selected condition (60 units, fixed before timing); each protocol keeps its own prompts, evidence budget, judgments, and extractions, and every unit is timed twice in interleaved order with the prefix cache reset beforehand, so only the reuse a protocol supports within a unit is credited. On this

sample and hardware the progressive schedule is 3–5% faster per condition than the single judgment (ratios 1.05, 1.04, and 1.03): the prefill it saves outweighs its additional judgment requests, while the HTML decoding shared by both protocols keeps the wall-clock difference well below the 22–28% by which the schedule's prefilled tokens fall short of the single judgment's on this engine (Table 14).

Table 16: Measured inference time per condition in seconds on vLLM 0.27.1 with generic prefix caching, one NVIDIA RTX PRO 6000 (96 GB): 20 documents per dataset with one condition each, two interleaved runs per protocol after warm-up, prefix cache reset before every unit; per-unit times are averaged over the two runs, then over the 20 units. The time covers all candidate judgments, incremental evidence processing, and HTML generation of accepted tables, and excludes model loading, upstream detection/OCR, and index construction. Ratio = single judgment / PCRC schedule; > 1 means the progressive schedule is faster.

Dataset	PCRC schedule + generic prefix cache (s)	Single judgment + generic prefix cache (s)	Ratio
FINCONTAB	62.8	65.7	1.05
PUBCONTAB	45.6	47.3	1.04
DEGCONTAB	59.9	61.9	1.03

A.4.6 BALANCE BETWEEN THE TWO FACTORS OF THE RELEVANCE SCORE

The relevance score of Eq. 1 is a product of a semantic and a spatial factor. A natural question is whether the spatial factor, being an exponential of a distance, can overwhelm the semantic one. This subsection reports the measured score distributions and a sweep over the decay rate.

What determines a factor's influence. For a multiplicative score, the quantity that matters is each factor's *spread*, the ratio between its typical high and low values. The spread is the multiplicative gap a factor can open between two candidate paragraphs, and therefore how much say it has in the final ranking; the absolute level only rescales all products equally and leaves the ranking unchanged. The exponential itself is not steep in this respect: the magnitude of the derivative of $\exp(-d)$ never exceeds 1 over the whole domain. What could in principle unbalance the two factors is the *scale* at which distance enters the exponential. The parameter-free treatment we adopt is to normalize the distance within each document, $\hat{d} = d/d_{\max} \in [0, 1]$, before applying the exponential, which confines $\exp(-\hat{d})$ to $[1/e, 1] \approx [0.368, 1]$ so the spatial factor cannot open an unbounded gap.

Measured distributions. Table 17 reports both distributions over all paragraph–table pairs of the three datasets. Two points follow. First, the semantic factor is not confined to a narrow band: it spans a 4.6–5.5× ratio between its 5th and 95th percentiles. Second, the two factors lie on essentially the same scale, and the spatial spread (2.2×) is the *smaller* of the two. The spatial factor therefore acts as a mild locality modulation rather than a dominating term. All results in the paper use this normalized form.

Sweeping the decay rate. To probe the boundary directly we replace the spatial factor with $\exp(-\lambda\hat{d})$ and vary λ, which widens or narrows its spread while leaving the semantic factor untouched. Table 18 reports both the resulting spread and the resulting quality. The two views agree. At $\lambda = 0.5$ the spatial factor is nearly muted, with a spread of about 1.5×. At $\lambda = 2$ the two distributions are almost exactly balanced (4.7–5.0× against 4.6–5.5× for the semantic factor). At $\lambda \geq 3$ the spatial factor takes the dominant role, with spreads of 10× and beyond. Quality is flat between $\lambda = 0.5$ and $\lambda = 1$, differing by at most 0.008 judgment F_1 on any dataset, and falls once λ exceeds 1, by 2.4–3.5 points at $\lambda = 4$. Cost moves in the opposite direction: prefill rises monotonically beyond the default, from 32.95/13.43/19.57k at $\lambda = 1$ to 40.93/17.18/25.37k at $\lambda = 4$ on FINCONTAB, PUBCONTAB, and DEGCONTAB, and the recomputation ratio follows the same upward trend. Settings in which the spatial factor dominates thus cost more and deliver less, so there is nothing to gain from pushing λ past the default.

A.4.7 BACKBONE GENERALITY

Claude Opus 4.5 (Anthropic, 2025), GPT-5.2 (OpenAI, 2025), and Qwen3-VL-Plus (Alibaba Cloud, 2026) are served through API endpoints that do not expose internal KV states. We therefore eval-

Table 17: Empirical distribution of the two factors of the relevance score $r = s_{\text{sem}} \cdot s_{\text{spa}}$, over all paragraph–table pairs of each dataset. Spread is the ratio q_{95}/q_{05}, i.e. the multiplicative gap a factor can open between two candidate paragraphs.

Factor	Dataset	q05	q25	median	q75	q95	Spread
$s_{\text{sem}} = \max\{0, \cos\}$	FINCONTAB	0.088	0.167	0.240	0.329	0.483	5.50×
	PUBCONTAB	0.133	0.247	0.342	0.448	0.608	4.56×
	DEGCONTAB	0.093	0.164	0.220	0.289	0.488	5.27×
$s_{\text{spa}} = \exp(-\hat{d})$	FINCONTAB	0.447	0.598	0.740	0.871	0.972	2.17×
	PUBCONTAB	0.434	0.584	0.732	0.869	0.973	2.24×
	DEGCONTAB	0.445	0.594	0.740	0.868	0.971	2.18×

Table 18: Decay-rate sweep on the normalized distance, $s_{\text{spa}} = \exp(-\lambda\hat{d})$. The three blocks report the spread of the spatial factor, the resulting quality, and the cost; † marks the default. Pre. is per-condition prefill in thousands and R.R. the recomputation ratio in percent.

λ	FIN	PUB	DEG
Spread of the spatial factor (q_{95}/q_{05})			
0.5	1.47	1.50	1.48
$1^{\dagger}$	2.17	2.24	2.18
2	4.73	5.02	4.76
3	10.27	11.24	10.38
4	22.33	25.18	22.65
semantic factor	*5.50*	*4.56*	*5.27*
Quality (F_1 / F_1-TEDS)			
0.5	0.909 / 0.722	0.903 / 0.752	0.899 / 0.793
$\mathbf{1}^{\dagger}$	**0.901 / 0.723**	**0.901 / 0.756**	**0.902 / 0.794**
2	0.894 / 0.716	0.896 / 0.748	0.893 / 0.786
3	0.881 / 0.702	0.884 / 0.734	0.876 / 0.771
4	0.866 / 0.684	0.869 / 0.719	0.878 / 0.758
Cost (Pre. (k) / R.R. (%))			
0.5	30.59 / 31.86	13.16 / 31.54	20.59 / 28.32
$\mathbf{1}^{\dagger}$	**32.95 / 28.29**	**13.43 / 29.02**	**19.57 / 24.35**
2	34.65 / 37.14	14.28 / 32.38	21.24 / 29.64
3	37.36 / 36.68	15.62 / 35.11	23.47 / 32.49
4	40.93 / 42.25	17.18 / 37.87	25.37 / 35.79

uate them only as single-shot baselines under the same input/output protocol and evaluation set as Qwen2.5-VL-7B. Since their KV states are unavailable, token-level cache comparison is not possible. None of these models was used to generate the benchmark conditions or labels. The condition layer for all three datasets was generated with Gemini 3.5 Flash (Google DeepMind, 2026) (Appendix A.1.6). To test PCRC on a stronger model, we run the full framework on GLM-4.5V (V Team et al., 2026), an open-weight model with accessible KV states. Table 19 reports both groups.

A.4.8 EFFECT OF SURROUNDING CONTEXT ON EXTRACTION

We further isolate whether surrounding text helps the extraction stage itself. For each table, we query Claude Opus 4.5, GPT-5.2, and Qwen3-VL-Plus under two otherwise identical input settings. Setting A gives the model the table image and unstructured table OCR; Setting B adds the top-ranked shared related paragraph and the top-2 exclusive related paragraphs of the relevance index. We focus on VLMs here because models that can directly consume both the table crop and related paragraph text are primarily vision-language models. Both settings use the same one-shot HTML formatting example. We evaluate the reconstructed HTML with cell-count weighted TEDS, and report $\Delta = \text{B} - \text{A}$ in Table 20.

The results show that related paragraphs are useful as auxiliary evidence, with the largest effect appearing when the table image is difficult. On the cleaner FINCONTAB and PUBCONTAB sources,

Table 19: Backbone generality. The three frontier models are served through API endpoints that do not expose internal KV states, so they are evaluated as single-shot baselines only and no cache-level cost is measurable (—). GLM-4.5V is an open-weight backbone on which the full PCRC contract is executed. Within the GLM-4.5V block, best per quality column is in **bold**; on Pre. and R.R., best between PCRC and PCRC w/o cache is in **bold**.

	Quality (F_1 / F_1-TEDS)			**Cost** (Pre. (k) / R.R. (%))		
Backbone / Method	FIN	PUB	DEG	FIN	PUB	DEG
Frontier VLMs, single-shot						
Claude Opus 4.5	0.938 / 0.859	0.939 / 0.861	0.967 / 0.848	—	—	—
GPT-5.2	0.955 / 0.846	0.942 / 0.871	0.923 / 0.849	—	—	—
Qwen3-VL-Plus	0.927 / 0.847	0.933 / 0.869	0.936 / 0.816	—	—	—
GLM-4.5V						
Single-shot	0.897 / 0.766	0.884 / 0.776	0.881 / 0.804	32.39 / —	13.15 / —	11.62 / —
PCRC	**0.951 / 0.842**	**0.952 / 0.863**	**0.958 / 0.877**	**39.83 / 30.78**	**15.95 / 30.70**	**19.96 / 22.32**
PCRC w/o cache	**0.951 / 0.842**	**0.952 / 0.863**	**0.958 / 0.877**	129.39 / 100	51.95 / 100	89.42 / 100

Table 20: Context ablation across three datasets and three frontier multimodal LLMs. FIN, PUB, and DEG denote FINCONTAB, PUBCONTAB, and DEGCONTAB, respectively. Setting A feeds the table image together with the unstructured table OCR. Setting B additionally provides the top-ranked shared related paragraph and the top-2 exclusive related paragraphs of the relevance index. The metric is TEDS, cell-count weighted.

		TEDS		
Dataset	**Model**	A	B	Δ
FIN	Claude Opus 4.5	0.653	0.661	+0.008
	GPT-5.2	0.674	0.678	+0.004
	Qwen3-VL-Plus	0.646	0.649	+0.003
PUB	Claude Opus 4.5	0.856	0.859	+0.003
	GPT-5.2	0.871	0.874	+0.003
	Qwen3-VL-Plus	0.835	0.841	+0.006
DEG	Claude Opus 4.5	0.555	0.705	+0.150
	GPT-5.2	0.665	0.761	+0.096
	Qwen3-VL-Plus	0.665	0.873	+0.208

context provides small positive extraction gains: all TEDS deltas are positive but below one point. On DEGCONTAB, the benefit is much stronger and consistent across models: related paragraphs raise TEDS by 9.6–20.8 points. This pattern matches the construction of DEGCONTAB: visual degradation, complex ruling lines, and multi-level headers make the crop alone less reliable, so related paragraphs can help recover table topic, variable meaning, measurement context, and header scope during reconstruction.

A.4.9 DISCUSSION

Unified Structure. The results show that query-conditioned table extraction should be treated as a coupled selection-and-reconstruction task rather than as conventional table extraction followed by filtering. Modular pipelines are vulnerable to intermediate errors: noisy extracted HTML can mislead the relevance judge, while missed relevant tables cannot be recovered by later reconstruction. In contrast, PCRC allows table selection and HTML reconstruction to share the same table-centered evidence, making the overall workflow more robust.

Context-aware Inference. Our results also highlight the importance of document context for table understanding. Surrounding paragraphs can clarify a table's semantic scope, which is essential for relevance judgment and can also support reconstruction when the crop alone is ambiguous. However, using context naively increases the prefill cost of VLM inference. PCRC addresses this tension through cache-aware inference: shared document evidence is reused across candidate tables, and

table-specific evidence is accumulated progressively only when needed. This suggests that future table extraction systems should move beyond isolated table recognition toward context-aware and computation-aware document interaction.