CLAIMCRAWL: CITATION-SUPERVISED RETRIEVAL AND RERANKING FOR PATENT PRIOR-ART SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Patent prior-art search requires matching the technical content of a claim to earlier patents expressed in different terminology and lengthy descriptions. Beyond finding relevant documents, a useful search system must place them within the small set a searcher can inspect. Our analysis shows that high recall at deep retrieval cutoffs can coexist with limited target coverage among the earliest results. This gap motivates task adaptation at both retrieval and ranking stages, with separate evaluation of their contributions. We present ClaimCrawl, a framework that uses claim-level examiner citations to supervise a dense retriever and a pointwise language-model reranker. The retriever learns from cited positives and complementary lexical hard negatives and cross-device negatives. The reranker uses claim-conditioned excerpts to predict structured citation-alignment labels and scores, prioritizing targets within a fixed candidate pool. We also construct PAR4PC-Retrieve, linking PANORAMA queries and citation targets to a frozen corpus of over seven million U.S. patents. Supervised retrieval improves Recall@5000 by 8.71 percentage points over the same unadapted encoder. Within fixed candidate pools, reranking improves Recall@20 by 14.40 points over the retriever's original ordering and 8.56 points over task-adapted Qwen3-Reranker-8B. Model comparisons and ablations examine the contributions of task adaptation, and a blinded human study assesses the usefulness of the returned shortlists. Together, these results support citation-supervised retrieval and reranking as complementary components of patent prior-art search. Code: `anonymous.4open.science/r/ClaimCrawl_code-BFBF/`.

1 INTRODUCTION

Patent prior-art search identifies earlier documents relevant to the technical features of a patent claim. Claims may combine features expressed under different terminology across long patent documents, making both retrieval and inspection demanding (Lupu & Hanbury, 2013; Jiang & Goetz, 2025; Shomee et al., 2025). A useful system must therefore recover relevant patents from a large collection and place them within the small set a searcher can inspect.

Retrieval methods range from lexical matching, as in BM25, to dense representations that capture semantic similarity (Robertson & Zaragoza, 2009; Karpukhin et al., 2020). Patent-specific encoders further exploit domain supervision: PaECTER trains a patent encoder with examiner-added citations (Ghosh et al., 2024); the patembed family uses multi-task training, with PatenTEB providing a benchmark for patent embeddings (Ayaou & Cavallucci, 2025); and QaECTER combines patent citation graphs with multi-view self-alignment (Djemmal et al., 2026).

Reranking enables closer comparison of queries and retrieved candidates. For long patent documents, Stamatis et al. (2024) combine lexical and semantic signals. In general text retrieval, RankGPT prompts LLMs to reorder candidate passages (Sun et al., 2023), Pairwise Ranking Prompting compares candidates in pairs (Qin et al., 2024), and FIRST derives a listwise ordering from the logits of the first generated identifier (Gangi Reddy et al., 2024). Within a fixed pool, however, reranking can promote an available target but cannot add a missing patent.

We study claim-level citation adaptation at two evaluation boundaries: corpus recovery and early ranking within shared candidate pools. The supervised crawler reaches 0.9365 Recall@5000 but

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.

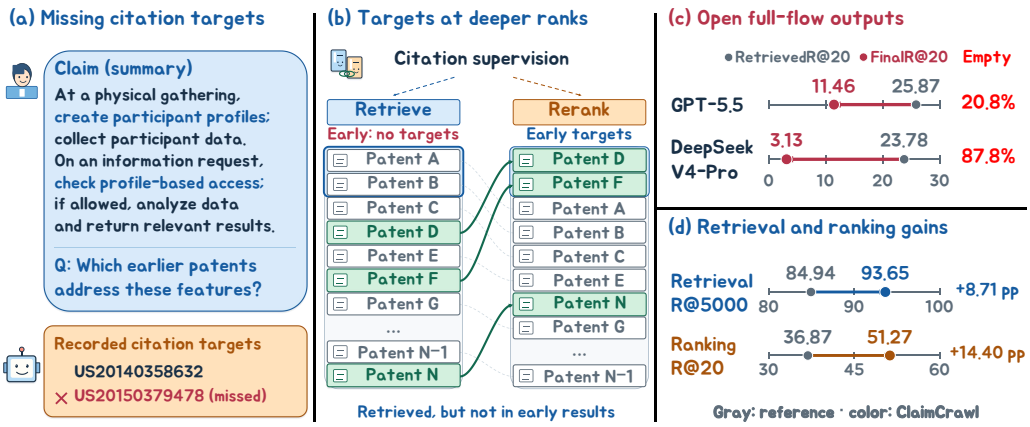

Figure 1: Retrieval and ranking bottlenecks. (a) A retrieval scenario with a real claim and its recorded citation targets. (b) Schematic promotion of retrieved targets from deeper ranks. (c) Retrieved/final Recall@20 and empty-output rates for open-output APIs on 2,896 queries; recall includes empty outputs. (d) Recall gains over the unadapted retriever and crawler ordering. Values are percentages; pp: percentage points. Axes in (d) are truncated.

0.3687 Recall@20. Reranking its leading 1,000 candidates raises Recall@20 to 0.5127, compared with 0.4271 for task-adapted Qwen3-Reranker-8B.

Raw-claim API tests of GPT-5.5 and DeepSeek V4-Pro (OpenAI, 2026; DeepSeek-AI, 2026) compare open configurations, which allow empty lists, with forced-nonempty output (Table 5). Final Recall@20 changes from 3.13% to 22.74% for DeepSeek and from 11.46% to 11.11% for GPT. Nonempty output does not ensure citation recovery. These complete configurations retrieve different lists; our primary ranking comparison fixes the pool and output size. Figure 1 illustrates the bottlenecks and open-output observations, using a PANORAMA claim and recorded targets (Lim et al., 2025).

With ClaimCrawl, claim-level examiner citations supervise two learning tasks: recovering cited patents from the corpus and prioritizing them within a bounded pool. The crawler learns claim–patent similarity from cited positives, lexical hard negatives, and batch comparisons. The pointwise selector processes claim-conditioned excerpts and predicts a citation-alignment label and score. We compare an encoder before and after adaptation, then compare rankers on identical pools. A separate inspection analysis measures the coverage–budget tradeoff under a fixed proxy.

PAR4PC-Retrieve joins PANORAMA claim-level supervision to official USPTO releases. Historical Full supports citation-based training and testing; Current Full supplies a separate human evaluation. Our contributions are:

- We adapt corpus retrieval and pointwise ranking with claim-level citation supervision, lexical hard negatives, cross-device comparisons, and claim-conditioned selector inputs.
- We construct PAR4PC-Retrieve, connecting claim-level citation targets to 7,039,592 U.S. patents for reproducible corpus-level retrieval evaluation.
- We measure adaptation and ordering gains through same-encoder and fixed-pool comparisons. Negative-source ablations and selector controls examine the training choices, while a 50-case human study assesses shortlist usefulness.

2 Related Work

Patent retrieval and task resources. Prior-art retrieval combines technical-language matching with examination-specific relevance. Earlier work combines BM25 retrieval with a BERT relevance gate (Stamatis, 2022); subsequent studies examine query formulation, neural reranking, and patent-family retrieval (Chikkamath et al., 2024; Stamatis et al., 2024; Ayaou et al., 2026;

Table 1: Selected patent benchmarks (PatenTEB: retrieval tasks). ✓: included; ×: absent from the indicated task.

Benchmark	Corpus search	Claim links
PatentMatch (Risch et al., 2021)	×	✓
ClaimCompare (Parikh & Dori-Hacohen, 2024)	×	×
PANORAMA–PAR4PC (Lim et al., 2025)	×	✓
DAPFAM (Ayaou et al., 2026)	✓	×
PatenTEB (Ayaou & Cavallucci, 2025)	✓	×
Sophia-Bench (Djemmal et al., 2026)	✓	×
FiNE-Patents (Knappich et al., 2026)	×	✓
PAR4PC-Retrieve	✓	✓

Corpus search uses a shared multi-patent collection. Claim links map individual claims to examiner-cited evidence. Our labels come from PANORAMA.

Chikkamath et al., 2026). PatentMatch provides claim–passage pairs (Risch et al., 2021); PANORAMA exposes examination trails and citation targets (Lim et al., 2025). FiNE-Patents links novelty prediction to passage evidence (Knappich et al., 2026), while PatentMind evaluates multi-aspect patent similarity (Yoo et al., 2026). PAR4PC-Retrieve connects existing claim-level citation labels to a shared corpus for retrieval and ranking evaluation.

Patent representations and dense retrieval. PaECTER learns representations from examiner citations (Ghosh et al., 2024); PatenTEB benchmarks patent embeddings (Ayaou & Cavallucci, 2025); and QaECTER combines citations with multiple patent views (Djemmal et al., 2026). Long-document alternatives include invention-graph encoders (Daniell et al., 2025) and Sparse Coverage's local semantic centers (Zuo et al., 2026). H-ProtoRAG couples hierarchical CPC modeling with retrieval (Elmi, 2026), while CPC-aware retrieval also uses LLM-assisted query expansion (Elmi et al., 2026). Beyond patents, MixGR uses complementary granularities (Cai et al., 2024), and Revela learns through cross-document language modeling (Cai et al., 2026). Our comparisons examine claim-level citation adaptation within a single-vector crawler and a separate pointwise selector.

Learned and LLM-based ranking. RankGPT and FIRST study listwise ranking (Sun et al., 2023; Gangi Reddy et al., 2024), while Qin et al. (2024) use pairwise prompts. Qwen3-Reranker scores query–document pairs using the relative likelihood of yes/no labels (Zhang et al., 2025). EcoRank controls ranking cost (Rashid et al., 2024); Rank1 distills reasoning traces (Weller et al., 2025); and Rank-R1 and ReasonRank learn reasoning-based ranking through reinforcement learning (Zhuang et al., 2026; Liu et al., 2026a). Our selector generates citation-alignment labels and scores. We compare against both unadapted and task-adapted Qwen3-Reranker, and use a separate yes/no-model control to distinguish output-policy effects from model comparisons.

Agentic search and evaluation. DeepResearcher and Agentic-R study learned search behavior (Zheng et al., 2025; Liu et al., 2026b). AstaBench controls tools and reports costs for scientific agents (Bragg et al., 2026); MC-Search evaluates intermediate search processes (Ning et al., 2026). We separate corpus retrieval, fixed-pool ranking, and raw-claim configurations, measuring patent citation recovery at each interface.

Table 1 positions our retrieval setting among existing patent benchmarks.

3 Task and Retrieval Collection

3.1 Claim-to-Patent Retrieval

Given a claim q_i and corpus $\mathcal{D}$, the task returns ranked patents. Recorded examiner citations $\mathcal{G}_i \subseteq \mathcal{D}$ serve as targets. With incomplete relevance labels, citation recovery need not capture all useful results (Buckley & Voorhees, 2004). Figure 2 illustrates claim summaries and recorded targets from PANORAMA (Lim et al., 2025). Targets link to full patent records; models process bounded text representations.

Citation recovery and usefulness are evaluated separately, with the latter assessed in the 50-case human study.
Here, corpus-level retrieval means that candidates must be found in the shared inventory rather than supplied as query–document pairs. The collection is frozen for reproducibility; the task does not assume access to every patent or every type of prior-art document.
For a ranked list $\mathcal{R}_i = (d_{i,1}, d_{i,2}, \ldots)$, we measure

$$\text{Recall}\,@K(q_i) = \frac{|\mathcal{R}_i^K \cap \mathcal{G}_i|}{|\mathcal{G}_i|}. \tag{1}$$

Here $\mathcal{R}_i^K$ contains the first $\min(K, |\mathcal{R}_i|)$ identifiers. Recall is macro-averaged across queries. MRR averages the first target's reciprocal rank in the returned list, with zero if absent. $\text{AllGold}\,@K$ counts queries with all targets in the top K. Evaluation covers the crawler's deeper lists and the selector's 100 outputs, emphasizing the first 20.

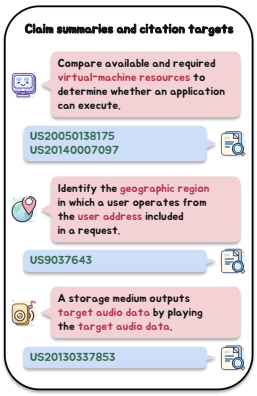

Figure 2: PANORAMA claim summaries and recorded citation targets.

3.2 Corpus Construction and Data Splits

PAR4PC-Retrieve connects PANORAMA PAR4PC queries and examiner-cited patents to a frozen search collection. Original PAR4PC supplies eight candidates per claim (Lim et al., 2025); our setting reuses its citation labels and searches for those targets in the shared corpus. We retain queries with U.S. publication targets and ingest official USPTO application and grant releases. PANORAMA target documents supplement missing official records where needed to retain an eligible target. Publication identifiers are normalized, and duplicate identifiers retain the most complete text record. Patent-family membership is metadata rather than the document identity key. Appendix A.3 details ingestion, duplicate handling, and query integrity.

Historical Full contains 7,039,592 patents published through December 31, 2024. The query split contains 53,942 training, 3,029 validation, and 2,896 test instances. A deterministic seed-42 sample of 500 validation queries selects model configurations and candidate depths before test reporting. Current Full adds 620,202 later records, reaching 7,659,794 patents through July 21, 2026. It supplies a separate evaluation of 50 post-2024 applications using human judgments rather than recorded-citation recall.

We retain PANORAMA's original PAR4PC split assignments, whose construction groups records by patent application (Lim et al., 2025). Distinct claim contexts remain separate instances within their assigned split. Publication identifiers define document identity and citation credit; family members remain separate documents. We apply no additional family grouping or per-query date filter. The shared corpus cutoff defines frozen-inventory citation recovery. Table 7 records split and identifier rules.

4 ClaimCrawl: Retrieval and Ranking

4.1 Overview and Candidate Handoff

To combine broad retrieval with more detailed candidate assessment, ClaimCrawl separates vector search from claim-conditioned ranking (Figure 3). The crawler searches the full corpus and returns 5,000 patents. Its leading 1,000 form the selector input, and the selector returns the best 100. This handoff controls the number of expensive claim–patent evaluations and supplies the same document set to competing rankers. The inspection analysis operates on at most the first 20 results after ranking is complete.

The two models are adapted separately from examiner citations. The crawler learns corpus-level similarity, while the selector predicts citation alignment for each claim–patent pair; its decoded label and score determine the final ranking.

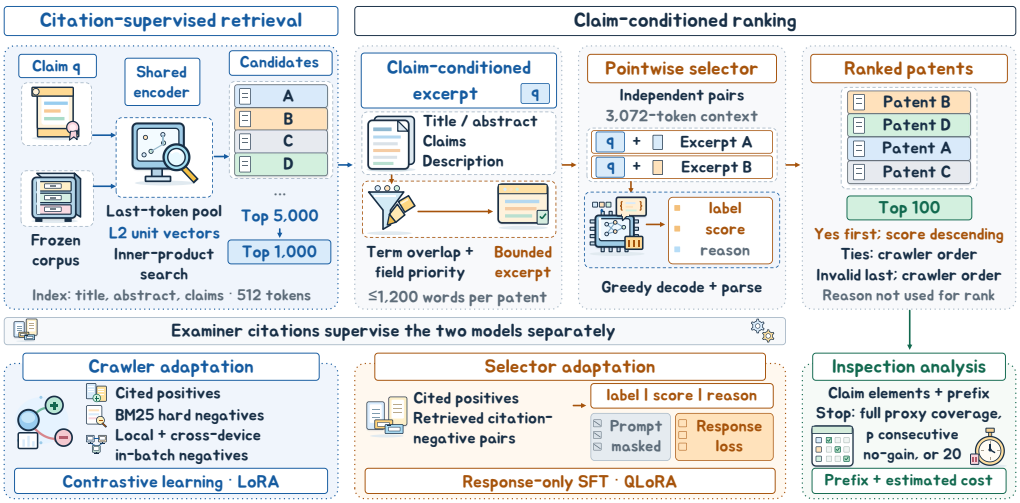

Figure 3: ClaimCrawl retrieves 5,000 patents, independently scores the leading 1,000, and returns 100. Citation supervision adapts the models separately. The inspection branch analyzes fixed-proxy coverage over at most 20 ranked patents (Appendix B.7). Colored examples are schematic.

4.2 Supervised Crawler for High Recall

To retrieve cited patents despite differences in terminology, we adapt Qwen3-Embedding-4B (Zhang et al., 2025) using LoRA (Hu et al., 2022) and claim–patent supervision. The encoder produces last-token representations normalized to unit length,

$$h_\phi(x) = \frac{\mathrm{Last}(\mathrm{Encoder}_\phi(x))}{\|\,\mathrm{Last}(\mathrm{Encoder}_\phi(x))\|_2}, \qquad s_\phi(q,d) = h_\phi(q)^\top h_\phi(d). \tag{2}$$

At indexing time, patent inputs concatenate title, abstract, and claims within 512 tokens. Inner-product search ranks the corpus by s_ϕ, using the same representation function for queries and documents.

Each loss term designates one cited patent $d_i^+ \in \mathcal{G}_i$ for claim q_i. With B paired examples and comparison set $\mathcal{B}_i$, the single-positive contrastive objective is

$$\mathcal{L}_{\mathrm{ret}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(s_\phi(q_i, d_i^+)/T)}{\sum_{d\in\mathcal{B}_i} \exp(s_\phi(q_i,d)/T)}, \qquad T = 0.03. \tag{3}$$

$\mathcal{B}_i$ contains d_i^+, gathered batch positives, and eight query-specific BM25 hard negatives (Robertson & Zaragoza, 2009). Explicit BM25 negatives exclude all recorded targets in $\mathcal{G}_i$; other batch positives are contrasted against the designated target. Cross-device gathering broadens comparisons, while lexical negatives challenge shared-word matching (Karpukhin et al., 2020; Xiong et al., 2021; Qu et al., 2021). Table 3 tests both sources; removing them retains local batch comparisons. Appendix B.2 distinguishes the single-positive objective from the full evaluation target set.

4.3 Task-Adapted Pointwise Selector

To move retrieved targets into the early ranking, we train a Qwen3.5-9B selector (Qwen Team, 2026) to predict a structured response for each claim–patent pair. Candidate text is assembled from title, abstract, claims, and description. Query-term overlap orders candidate passages with preferences for title, abstract, and claims, and the excerpt is bounded to 1,200 words before tokenization. The 9B model uses a 3,072-token context. This gives each patent a bounded claim-conditioned input, independently of other candidates' document lengths.

For passage u and token set $V(\cdot)$, the excerpt construction orders passages by

$$a(u,q) = \frac{|V(u)\cap V(q)|}{\max(1,|V(q)|)} + w(u), \tag{4}$$

Table 2: Corpus retrieval (a) and fixed-pool ranking (b) on 2,896 queries (metrics as fractions). Bold: panel best, including ties; red: selected; †: Validation-500 selection.

(a) Corpus recovery: full corpus → 5,000 patents					
Method	**R@20↑**	**R@100↑**	**R@1000↑**	**R@5000↑**	**MRR↑**
Static claim elements	0.1742	0.3023	0.4462	0.4573	0.1334
BM25	0.2477	0.3981	0.6267	0.7875	0.1892
Dense-only	0.2897	0.4726	0.7542	0.8936	0.1693
BGE-M3 (zero-shot)	0.2389	0.3618	0.5811	0.7287	0.1514
PaECTER	0.2032	0.3450	0.5576	0.7290	0.1320
patembed-large	0.2436	0.4056	0.6482	0.8094	0.1522
Qwen3-Embedding 0.6B (zero-shot)	0.2999	0.4835	0.7222	0.8371	0.1836
Qwen3-Embedding 0.6B (trained)	0.3389	0.5257	0.7811	0.9195	0.1976
Qwen3-Embedding 4B (zero-shot)	0.3115	0.4862	0.7275	0.8494	0.1989
ClaimCrawl crawler 4B†	**0.3687**	0.5724	**0.8127**	0.9365	**0.2136**
Qwen3-Embedding 8B (zero-shot)	0.3157	0.4924	0.7185	0.8467	0.1993
Qwen3-Embedding 8B (trained)	0.3681	**0.5798**	0.8094	**0.9366**	0.2050
(b) Fixed-pool ordering: same 1,000 candidates → 100 patents					
Method	**R@10↑**	**R@20↑**	**R@50↑**	**R@100↑**	**MRR↑**
Crawler input order	0.3019	0.3687	0.5119	0.5724	0.2136
GPT-5.5 forced selector	0.3019	0.3687	0.5119	0.5724	0.2275
DeepSeek V4-Pro forced selector	0.3383	0.3474	0.5102	0.5584	0.2405
Qwen3-Reranker-8B (zero-shot)	0.3221	0.3981	0.4828	0.5341	0.2778
Qwen3-Reranker-8B (LoRA)	0.3627	0.4271	0.5667	0.6218	0.2982
Qwen3.5-9B (no LoRA)	0.1551	0.1915	0.2540	0.3200	0.0949
ClaimCrawl selector 9B†	**0.4333**	**0.5127**	**0.6364**	**0.7332**	**0.3438**
Qwen3.5-4B (LoRA)	0.4263	**0.5127**	**0.6364**	**0.7332**	0.3287
Llama-3.1-8B (LoRA)	0.4233	**0.5127**	**0.6364**	0.7301	0.3381
Ministral-8B (LoRA)	0.3947	0.4572	0.5700	0.6707	0.3159

Here $w(u)$ is 1 for title or abstract, 0.25 for claims, and 0 for description. Ties preserve source order; ranked passages are concatenated and truncated to the word budget. This deterministic rule allocates context without an additional trained component. The crawler embeds each patent once; the selector chooses passages for the particular claim.

The response contains a yes/no citation-alignment label $\hat{y}$, a score $s \in [0, 1]$, and a short reason field. The training set has 120,000 pairs, with 25,626 recorded-citation positives and 94,374 retrieved citation-negative examples. Their serialized targets assign `true`/0.95 and `false`/0.05, respectively. We optimize the response-token likelihood

$$\mathcal{L}_{\text{sel}} = -\frac{1}{Z} \sum_{(q,d,z)\in\mathcal{S}} \sum_{t=1}^{|z|} \log p_\theta(z_t \mid x(q,d), z_{<t}), \tag{5}$$

where $x(q, d)$ is the prompt, z the serialized response, and $Z = \sum_{(q,d,z)\in\mathcal{S}} |z|$ the number of supervised tokens. Prompt tokens are masked. Training supervises response generation, including the label-associated score. Fixed reason strings encode training labels rather than document-grounded explanations.

4.4 Adaptation and Deterministic Inference

QLoRA adapts the language model while keeping its quantized backbone fixed (Dettmers et al., 2023). Greedy decoding produces one response per candidate. Ranking uses the predicted label first, then descending score, with crawler rank breaking ties. Scores provide within-label ordering, not calibrated relevance probabilities. The yes/no controls vary ordering on one checkpoint; comparison with the structured selector also changes training targets.

Citation-negative candidates remain eligible, malformed responses follow valid ones in crawler order, and the top 100 are returned. Reasons do not affect ranking. Even when all valid decisions

are negative, the selector returns 100 ordered candidates. The two models are adapted separately; Appendices B.3 and B.6 detail training, and Algorithm 1 specifies inference.

4.5 Post-Ranking Inspection Analysis

A separate analysis measures claim-element coverage along the completed ranking using fixed proxy decisions. Inspection stops when all elements are covered, after p consecutive patents add no coverage, or at 20 patents. Patience counts no-gain steps, so complete coverage can end inspection before p patents. This analysis holds retrieval and ranking fixed and measures proxy coverage against inspection budget (Appendix B.7).

5 Experiments

5.1 Setup and Comparison Boundaries

Citation-based comparisons use the 2,896-query test and Historical Full. Validation-500 selects the crawler, selector, and candidate depth.

Corpus-search baselines include BM25 (Robertson & Zaragoza, 2009), BGE-M3 (Chen et al., 2024), PaECTER (Ghosh et al., 2024), patembed (Ayaou & Cavallucci, 2025), and Qwen3-Embedding (Zhang et al., 2025). Fixed-pool rankers receive identical 1,000-candidate pools and return 100 patents. They include Qwen3-Reranker (Zhang et al., 2025), Qwen3.5 (Qwen Team, 2026), Llama-3.1 (Grattafiori et al., 2024), Ministral (Mistral AI Team, 2024), and prompted GPT-5.5 and DeepSeek V4-Pro selectors (OpenAI, 2026; DeepSeek-AI, 2026).

In comparison tables, bold marks the best value within each comparison group, including ties, according to the metric direction. Red identifies selected configurations. Starred rows are excluded from controlled depth comparisons and best-value highlighting.

We report query-macro recall and MRR. Main learned runs use seed 42; three-seed summaries assess training variation. Named encoder pairs isolate task adaptation with the backbone and input format held fixed. Appendix B records training settings, validation choices, inference interfaces, and output handling for the corpus-search and ranking experiments.

5.2 Main Results and Model Applicability

Table 2(a) separates task adaptation from encoder scale. Adapting the same 4B encoder raises Recall@1000 from 0.7275 to 0.8127 and Recall@5000 from 0.8494 to 0.9365. The latter is 4.29 percentage points above Dense-only (0.8936). The selected 4B run has higher Recall@1000 than trained 8B (0.8094), while their Recall@5000 values are close (0.9365 versus 0.9366). Selection follows validation performance, not uniform dominance across test metrics.

Panel (b) compares ranking configurations over the same 1,000 candidates. The 9B selector reaches 0.5127 Recall@20: 14.40 percentage points above crawler order and 8.56 above task-adapted Qwen3-Reranker-8B. Adapted 4B and Llama-3.1-8B match its Recall@20 and Recall@50, while MRR distinguishes first-target placement. Validation-500 selects 9B by MRR after a Recall@20 tie (Table 8). The same-9B controls in Table 3 examine adaptation and output policy.

5.3 Training Choices and Candidate Budgets

Table 3 answers two different questions. Panel (a) varies BM25 hard negatives and cross-device negatives for the 4B crawler; the unadapted encoder is a separate training reference. Panel (b) contrasts the trained 9B selector with crawler order, the same backbone without LoRA, and a trained yes/no model under two ordering policies.

For ranking, Recall@20 is 36.87% without reranking, 19.15% without task adaptation, and 51.27% for the full selector. The two yes/no rows share a checkpoint and candidate scores: one sorts by label likelihood; the other groups positive decisions first and retains crawler order within groups. Their difference tests output policy. Comparing either row with the structured selector changes the

training target as well as the ranking rule, so it evaluates the complete design. Additional metrics appear in Table 10.

Table 3: Ablations on 2,896 queries. Recall (%); MRR (fraction). Bold: panel best; red: full configuration. Panel (b) uses 1,000 candidates and 100 outputs; structured versus yes/no comparisons change training targets.

(a) Crawler: adaptation and negative sources				
Configuration	**R@100 ↑**	**R@1000 ↑**	**R@5000 ↑**	**MRR ↑**
Full crawler: both negative sources	**57.24**	**81.27**	**93.65**	**0.2136**
Without BM25 hard negatives	49.17	73.15	83.32	0.1924
Without cross-device negatives	54.11	80.17	92.27	0.2022
Without both added sources	43.86	69.22	84.28	0.1874
Unadapted 4B encoder	48.62	72.75	84.94	0.1989
(b) Selector: adaptation and output policy				
Configuration	**R@20 ↑**	**R@100 ↑**		**MRR ↑**
Full selector: trained label + score	**51.27**	**73.32**		**0.3438**
Without reranking (crawler order)	36.87	57.24		0.2136
Without task adaptation (no LoRA)	19.15	32.00		0.0949
Yes/no model: rank by label likelihood	15.97	52.95		0.1172
Yes/no model: gate, then crawler order	42.71	52.43		0.3371

Removing BM25 negatives reduces Recall@5000 from 93.65% to 83.32%, whereas removing cross-device negatives gives 92.27%. Removing both added sources leaves local batch negatives and gives 84.28%, above the 83.32% obtained without BM25 negatives alone. Their contributions therefore depend on the training configuration. For the selector, the yes/no gate improves early recall over likelihood sorting but remains below the structured design. The full comparison measures the combined choice of training targets and ordering rule.

Across three training seeds, crawler Recall@5000 is 0.9333 ± 0.0040 and selector Recall@20 is 0.5080 ± 0.0072 (mean $\pm$ standard deviation). Table 11 lists each run.

Candidate and inspection budgets. The controlled depth comparison uses 500 and 1,000 selector inputs with 100 outputs (Table 10(a)). Recall@20 remains 51.27%, while Recall@100 rises from 70.54% to 73.32%. Validation selects depth by Recall@20, then Recall@100 and MRR (Table 8). The larger pool improves coverage of the full output even when the first 20 results recover the same target fraction. At 100 inputs, Recall@100 and All-Gold@100 measure coverage of the entire candidate set: both are unchanged by reordering and need not equal one. We retain the reported 100-input configuration separately ($*$); Appendix A.2 explains its candidate-set discrepancy with the crawler reference.

The inspection sweep in Table 4 uses the same ranking and proxy for every policy. Validation-500 selects the policy with the highest estimated token savings subject to an unsupported-element rate no higher than the all-20 reference. Among the tested patience values, 7 and 10 satisfy this constraint; 7 saves more tokens. Appendix B.8 gives the validation values.

At $p = 7$, mean inspection is 6.44 patents and proxy coverage is 75.43%, retaining 94.3% of the all-20 coverage (80.00%). Estimated token savings reach 61.8%. Increasing patience to 10 raises coverage to 77.14% but lowers savings to 52.5%. Thus, 7 is the validation-selected budget tradeoff, rather than the policy with maximum coverage. Savings measure estimated inspection tokens, not total pipeline latency.

Table 4: Inspection budgets under a fixed proxy. Saved: estimated tokens versus all 20; red: validation choice.

Policy	**Mean inspected↓**	**Coverage (%)↑**	**Saved (%)↑**
All 20	20.00	80.00	0.0
$p = 2$	2.85	62.29	84.9
$p = 3$	3.75	68.00	78.7
$p = 5$	5.06	69.71	71.5
$p = 7$	6.44	75.43	61.8
$p = 10$	7.96	77.14	52.5

5.4 RAW-CLAIM API SEARCH AND HUMAN EVALUATION

Table 5 evaluates APIs given only a claim, with open or forced-nonempty output. Table 2(b) instead supplies a fixed candidate pool and requires 100 outputs. Table 6 separately measures human-assessed usefulness on 50 later cases; its success rate is not citation recall.

Table 5: Raw-claim API search on 2,896 queries. Metrics include empty lists; red: empty counts.

		Retrieved	Final output						
Model	**Output**	**R@20↑**	**R@10↑**	**R@20↑**	**R@50↑**	**R@100↑**	**All-Gold@100↑**	**MRR↑**	**Empty↓**
GPT-5.5	Open	0.2587	0.1146	0.1146	0.1146	0.1146	0.0833	0.1458	603
GPT-5.5	Forced	0.2274	0.1042	0.1111	0.1111	0.1111	0.0833	0.1053	0
DeepSeek V4-Pro	Open	0.2378	0.0313	0.0313	0.0313	0.0313	0.0208	0.0417	2543
DeepSeek V4-Pro	Forced	0.2448	0.1979	0.2274	0.2274	0.2274	0.1458	0.1948	0

Open output yields 603 and 2,543 empty lists for GPT-5.5 and DeepSeek V4-Pro, respectively; forced output yields none. Final Recall@20 rises from 3.13% to 22.74% for DeepSeek but changes from 11.46% to 11.11% for GPT. Nonempty output alone therefore does not ensure citation recovery. Retrieved recall exceeds final recall in all four configurations, and empty lists contribute zero recall. Open and forced rows jointly vary retrieval and output behavior, so their differences describe complete configurations. Empty counts record the outcome without identifying whether it arose from abstention, requests, or parsing. Appendix B.5 distinguishes these API tests from browser-based search products.

The manual study evaluates 50 post-2024 applications from Current Full, excluded from PANORAMA splits (Lim et al., 2025). It assesses technical usefulness beyond citation matching. Candidates are pooled, de-duplicated, and randomized for blinded review (Appendix D.3).

Missing, duplicate, self-matching, or unresolved outputs retain their ranks and count as rejected. Appendix D.3 supplies the full manual protocol.

Three reviewers independently assess each pooled candidate with method identity and rank hidden, then jointly re-inspect disagreements. Success requires at least five top-20 patents to receive final unanimous acceptance as relevant or technically helpful. This consensus endpoint differs from independent initial agreement. API rows use forced full-flow configurations.

Table 6: Human evaluation on 50 later cases. Success: $\geq$5 unanimous acceptances in the top 20.

Method	**Success↑**
BM25	11/50 (22%)
BGE-M3 zero-shot	13/50 (26%)
Dense-only	15/50 (30%)
DeepSeek V4-Pro forced	17/50 (34%)
GPT-5.5 forced	22/50 (44%)
ClaimCrawl	**29/50 (58%)**

Table 6 reports 29 successful cases for ClaimCrawl, 22 for GPT-5.5, and 17 for DeepSeek V4-Pro. Methods retain their original top-20 positions. Acceptance requires claim relevance or substantial technical assistance beyond topical overlap. The five-patent threshold measures whether a shortlist supplies several useful leads. These counts describe usefulness on the 50 cases; Appendix D.4 gives uncertainty intervals without inferring a paired significance result.

Appendix D.2 specifies how per-query lists would localize retrieval, handoff, and final-output losses.

6 CONCLUSION

ClaimCrawl uses claim-level citation supervision to improve retrieval and ranking over PAR4PC-Retrieve's frozen patent corpus. Same-encoder comparisons measure citation-recovery gains from adaptation; fixed-pool comparisons measure early ordering while preserving candidate membership. Negative-source and selector controls characterize the training choices, and the later-application study provides descriptive evidence of shortlist usefulness.

AI USE STATEMENT

Generative AI assisted with literature search and reference checking, drafting and translation, figure and code editing and checking, and discussion of experimental design and results. The authors take responsibility for the final content.

ETHICS STATEMENT

ClaimCrawl retrieves and prioritizes patent documents for inspection. Recorded examiner citations are incomplete relevance labels, and the corpus covers U.S. patents rather than the entire prior-art universe. The reported scores therefore do not establish legal validity or search completeness. The manual study provides a preliminary evaluation of early-list utility; its protocol and interpretation are described in Appendix D.3.

REPRODUCIBILITY STATEMENT

The benchmark specifies corpus snapshots, splits, candidate pools, and evaluation endpoints. Appendix B gives configuration selection, training targets, prompts, and implementation settings; Appendix D.3 specifies manual evaluation. Additional results report seed variability and case-level uncertainty.

REFERENCES

Iliass Ayaou and Denis Cavallucci. PatenTEB: A comprehensive benchmark and model family for patent text embedding. arXiv preprint arXiv:2510.22264, 2025. URL https://arxiv.org/abs/2510.22264.

Iliass Ayaou, Denis Cavallucci, and Hicham Chibane. DAPFAM: A domain-aware family-level dataset to benchmark cross domain patent retrieval. *Array*, 29:100720, 2026. doi: 10.1016/j.array.2026.100720.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi Mishra, Sergey Feldman, Dany Haddad, Jena Hwang, Peter Jansen, Varsha Kishore, Bodhisattwa Prasad Majumder, Aakanksha Naik, Sigal Rahamimov, Kyle Richardson, Amanpreet Singh, Harshit Surana, Aryeh Tiktinsky, Rosni Vasu, Guy Wiener, Chloe Anastasiades, Stefanus Candra, Jason Dunkelberger, Daniel Emery, Rob Evans, Malachi Hamada, Regan Huff, Rodney Kinney, Matt Latzke, Jaron Lochner, Ruben Lozano-Aguilera, Ngoc-Uyen Nguyen, Smita Rao, Amber Tanaka, Brooke Vlahos, Peter Clark, Doug Downey, Yoav Goldberg, Ashish Sabharwal, and Daniel Weld. AstaBench: Rigorous benchmarking of AI agents with a scientific research suite. In *The Fourteenth International Conference on Learning Representations*, pp. 110136–110223, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/b2ce9568dbb559aefc8c98ca5b5314ce-Abstract-Conference.html.

Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 25–32. Association for Computing Machinery, 2004. doi: 10.1145/1008992.1009000.

Fengyu Cai, Xinran Zhao, Tong Chen, Sihao Chen, Hongming Zhang, Iryna Gurevych, and Heinz Koeppl. MixGR: Enhancing retriever generalization for scientific domain through complementary granularity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10369–10391. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.579. URL https://aclanthology.org/2024.emnlp-main.579/.

Fengyu Cai, Tong Chen, Xinran Zhao, Sihao Chen, Hongming Zhang, Sherry Wu, Iryna Gurevych, and Heinz Koeppl. Revela: Dense retriever learning via language modeling. In *International Conference on Learning Representations*, pp. 1372–1395, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0343104ddbfc48f35f06aaae88980e48-Abstract-Conference.html.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2318–2335. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.137. URL https://aclanthology.org/2024.findings-acl.137/.

Renukswamy Chikkamath, Deepak Rastogi, Mahesh Maan, and Markus Endres. Is your search query well-formed? a natural query understanding for patent prior art search. *World Patent Information*, 76:102254, 2024. doi: 10.1016/j.wpi.2023.102254.

Renukswamy Chikkamath, Linda Andersson, and Markus Endres. Rethinking patent retrieval with language models: Toward scalable and efficient search. *World Patent Information*, 84:102433, 2026. doi: 10.1016/j.wpi.2026.102433.

Krzysztof Daniell, Igor Buzhinsky, and Sebastian Björkqvist. Efficient patent searching using graph transformers. In *Proceedings of the 6th Workshop on Patent Text Mining and Semantic Technologies*, volume 4062 of *CEUR Workshop Proceedings*, pp. 51–59, 2025. URL https://ceur-ws.org/Vol-4062/paper5.pdf.

DeepSeek-AI. DeepSeek V4 preview release. Official API documentation, 2026. URL https://api-docs.deepseek.com/news/news260424/.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. doi: 10.52202/075280-0441. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html.

Younes Djemmal, You Zuo, Kim Gerdes, and Kirian Guiller. Citation-driven multi-view training for patent embeddings: QaECTER and sophia-bench. arXiv preprint arXiv:2604.22897, 2026. URL https://arxiv.org/abs/2604.22897.

Zahra Elmi. H-ProtoRAG: A hierarchical prototype-based retrieval-augmented framework for multilingual CPC patent classification and prior-art retrieval. *Information Processing & Management*, 63(7):104861, 2026. doi: 10.1016/j.ipm.2026.104861.

Zahra Elmi, Nursena Aydın, Süreyya Engin, Rümeysa Besli, and Soheila Elmi. CPC-aware neural patent retrieval with LLM-assisted query expansion. *Journal of Data, Information and Management*, 8(3):153–169, 2026. doi: 10.1007/s42488-026-00165-8.

Revanth Gangi Reddy, JaeHyeok Doo, Yifei Xu, Md Arafat Sultan, Deevya Swain, Avirup Sil, and Heng Ji. FIRST: Faster improved listwise reranking with single token decoding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8642–8652. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.491. URL https://aclanthology.org/2024.emnlp-main.491/.

Mainak Ghosh, Michael E. Rose, Sebastian Erhardt, Erik Buunk, and Dietmar Harhoff. PaECTER: Patent-level representation learning using citation-informed transformers. arXiv preprint arXiv:2402.19411, 2024. URL https://arxiv.org/abs/2402.19411.

Aaron Grattafiori et al. The Llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024. URL https://arxiv.org/abs/2407.21783.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Lekang Jiang and Stephan M. Goetz. Natural language processing in the patent domain: A survey. *Artificial Intelligence Review*, 58(7):214, 2025. doi: 10.1007/s10462-025-11168-z.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Valentin Knappich, Anna Hätty, Simon Razniewski, and Annemarie Friedrich. Is it novel and why? fine-grained patent novelty prediction based on passage retrieval. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 845–856. Association for Computing Machinery, 2026. doi: 10.1145/3805712.3809576.

Hyunseung Lim, Sooyohn Nam, Sungmin Na, Ji Yong Cho, June Yong Yang, Hyungyu Shin, Yoonjoo Lee, Juho Kim, Moontae Lee, and Hwajung Hong. PANORAMA: A dataset and benchmarks capturing decision trails and rationales in patent examination. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4aab82c8d6b77c0b6b010145c1bfcdd3-Abstract-Datasets_and_Benchmarks_Track.html.

Wenhan Liu, Xinyu Ma, Weiwei Sun, Yutao Zhu, Yuchen Li, Dawei Yin, and Zhicheng Dou. ReasonRank: Empowering passage ranking with strong reasoning ability. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22007–22031. Association for Computational Linguistics, July 2026a. doi: 10.18653/v1/2026.acl-long.1005. URL https://aclanthology.org/2026.acl-long.1005/.

Wenhan Liu, Xinyu Ma, Yutao Zhu, Yuchen Li, Daiting Shi, Dawei Yin, and Zhicheng Dou. Agentic-R: Learning to retrieve for agentic search. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 15987–16005. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.findings-acl.785. URL https://aclanthology.org/2026.findings-acl.785/.

Mihai Lupu and Allan Hanbury. Patent retrieval. *Foundations and Trends in Information Retrieval*, 7(1):1–97, 2013. doi: 10.1561/1500000027.

Mistral AI Team. Ministral-8B-Instruct-2410 model card. Hugging Face model repository, 2024. URL https://huggingface.co/mistralai/Ministral-8B-Instruct-2410.

Xuying Ning, Dongqi Fu, Tianxin Wei, Mengting Ai, Jiaru Zou, Ting-Wei Li, Hanghang Tong, Yada Zhu, Hendrik Hamann, and Jingrui He. MC-Search: Evaluating and enhancing multimodal agentic search with structured long reasoning chains. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=JEGDp1E4OH.

Soichi Onozuka and Takaaki Ohnishi. Dynamic search depth allocation for patent prior art search: An empirical study of retrieval limitations in 64-dimensional embeddings. In *2026 International Conference on Semantic Computing (ICSC)*, pp. 416. IEEE, 2026. doi: 10.1109/ICSC67292.2026.00069.

OpenAI. Introducing GPT-5.5. Official model release, 2026. URL https://openai.com/index/introducing-gpt-5-5/.

Arav Parikh and Shiri Dori-Hacohen. Claimcompare: A data pipeline for evaluation of novelty destroying patent pairs. In *Proceedings of the 5th Workshop on Patent Text Mining and Semantic Technologies*, volume 3775 of *CEUR Workshop Proceedings*, pp. 61–66, 2024. URL https://ceur-ws.org/Vol-3775/paper5.pdf.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1504–1518. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-naacl.97. URL https://aclanthology.org/2024.findings-naacl.97/.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5835–5847. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.466. URL https://aclanthology.org/2021.naacl-main.466/.

Qwen Team. Qwen3.5-9B model card. Hugging Face model repository, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Muhammad Rashid, Jannat Meem, Yue Dong, and Vagelis Hristidis. EcoRank: Budget-constrained text re-ranking using large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13049–13063. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.773. URL https://aclanthology.org/2024.findings-acl.773/.

Julian Risch, Nicolas Alder, Christoph Hewel, and Ralf Krestel. PatentMatch: A dataset for matching patent claims & prior art. In *Proceedings of the 2nd Workshop on Patent Text Mining and Semantic Technologies*, volume 2909 of *CEUR Workshop Proceedings*, pp. 40–44, 2021. URL https://ceur-ws.org/Vol-2909/paper5.pdf.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 4(1–2):1–174, 2009. doi: 10.1561/1500000019.

Homaira Huda Shomee, Zhu Wang, Sathya N. Ravi, and Sourav Medya. A survey on patent analysis: From NLP to multimodal AI. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8545–8561. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.419. URL https://aclanthology.org/2025.acl-long.419/.

Vasileios Stamatis. End to end neural retrieval for patent prior art search. In *Advances in Information Retrieval*, pp. 537–544. Springer, 2022. doi: 10.1007/978-3-030-99739-7_66.

Vasileios Stamatis, Michail Salampasis, and Konstantinos Diamantaras. A novel re-ranking architecture for patent search. *World Patent Information*, 78:102282, 2024. doi: 10.1016/j.wpi.2024.102282.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14918–14937. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.923. URL https://aclanthology.org/2023.emnlp-main.923/.

Orion Weller, Kathryn Ricci, Eugene Yang, Andrew Yates, Dawn Lawrie, and Benjamin Van Durme. Rank1: Test-time compute for reranking in information retrieval. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Pg0PAvbhGv.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=zeFrfgyZln.

Yongmin Yoo, Qiongkai Xu, and Longbing Cao. PatentMind: A multi-aspect reasoning graph for patent similarity evaluation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 14947–14964. Association for Computational Linguistics, July 2026. doi: 10.18653/v1/2026.findings-acl.735. URL https://aclanthology.org/2026.findings-acl.735/.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. arXiv preprint arXiv:2506.05176, 2025. URL https://arxiv.org/abs/2506.05176.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. DeepResearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 414–431. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.22. URL https://aclanthology.org/2025.emnlp-main.22/.

Shengyao Zhuang, Xueguang Ma, Zheng Yao, Shuai Wang, Bevan Koopman, Jimmy Lin, and Guido Zuccon. Rank-R1: Enhancing reasoning in LLM-based document rerankers via reinforcement learning. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 4419–4425. Association for Computing Machinery, 2026. doi: 10.1145/3805712.3809961.

You Zuo, Kim Gerdes, Éric de la Clergerie, and Benoît Sagot. Sparse Coverage: Semantic center representations for patent prior-art retrieval. arXiv preprint arXiv:2608.16918, 2026. URL https://arxiv.org/abs/2608.16918.

This appendix provides evaluation conventions, validation-based configuration selection, implementation details, evaluation protocols, and supplementary diagnostics.

A Data Construction and Evaluation Boundaries

A.1 Metric and Aggregation Conventions

Citation-based Recall@K is the per-query fraction of recorded examiner citations retrieved by rank K, macro-averaged across queries. MRR is the macro-average reciprocal rank of the first recorded citation in the evaluated returned list, with zero assigned when none is present. The list contains up to 5,000 crawler results, 100 fixed-pool outputs, or the available raw-claim outputs; MRR therefore has no @K suffix. All-Gold@K is the fraction of queries for which every recorded citation appears by rank K. It is a secondary completeness diagnostic, not an estimate of exhaustive legal relevance.

For raw-claim full flow, retrieved Recall@20 uses the retrieved list, whereas selected Recall@20 uses the final returned list. Empty counts queries with no selected patent. Open configurations allow empty final lists; forced-nonempty configurations have zero recorded empty lists. These output conditions do not identify the cause of an empty list or establish identical retrieved candidates across policies.

The manual endpoint is case-level success. A case succeeds when all three reviewers unanimously accept at least five of a method's top 20 patents under the protocol below. Proxy coverage is computed over each query's claim elements and then averaged across queries. Coverage retention divides a policy's average proxy coverage by the all-top-20 reference. Inspection savings and estimated token savings use the same reference. Unsupported denotes the rate of returned claim elements lacking support under the fixed proxy. It concerns returned elements, whereas coverage concerns all query elements; the two rates are not complements. It also differs from the fraction of inspection steps that add no new coverage.

A.2 Comparable System Boundaries

Crawler comparisons vary ranking over the same Historical Full corpus. Fixed-pool selector and reranker comparisons hold the leading 1,000 crawler candidates and the 100-patent output size fixed. Table 10(a) reports candidate depths, with the comparison boundary for 100 inputs specified below. Stopping comparisons use one ranking and proxy and vary patience. Raw-claim configurations change retrieval and final selection together and are reported separately.

These boundaries also govern ties and missing outputs. Crawler order resolves exact local-selector ties. Fixed-pool API outputs are restricted to supplied identifiers, de-duplicated, and completed in crawler order when fewer than 100 valid identifiers remain. Open raw-claim full flow preserves empty final lists. The GPT-5.5 fixed-pool row matches crawler Recall@10, @20, @50, and @100 but differs in MRR. Equal macro recall can coexist with different target ranks within those cutoffs. This row evaluates the complete API-and-completion interface; the aggregate scores alone do not attribute its behavior to generation or completion.

Candidate-set consistency. When all m unique input candidates are retained, reordering preserves both Recall@m and All-Gold@m. At 100 inputs and 100 outputs, both metrics describe candidate-set coverage and may be below one when targets are absent. They remain meaningful for comparisons across candidate depths, where the set available for top-100 selection changes.

For an identical 100-patent input and output set, both metrics must equal their values in crawler order. The reported 100-input configuration differs from the crawler reference: validation Recall@100 is 0.6273 versus 0.5234; test Recall@100 is 0.5058 versus 0.5724, and test All-Gold@100 is 0.3117 versus 0.3441. These differences preclude treating that row as a permutation of the reference crawler prefix without a per-query pool alignment. Tables 8(c) and 10(a) preserve all reported 100-input metrics and mark this comparison boundary with $*$. Controlled depth conclusions and best-value highlighting use the 500- and 1,000-input settings. Recall@100 and All-Gold@100 remain valid coverage measures at 100 inputs.

A.3 Corpus Construction, Provenance, and Integrity

Table 7 combines reported construction counts with the normalization rules used by the corpus pipeline. A deterministic text-derived identifier is used only for a source record without a publication identifier and never overrides a valid identifier. For evaluation, a prediction row's gold set takes precedence when present; identifier-only fallback matching does not collapse distinct claim instances. Together, these invariants keep corpus membership fixed across crawler runs, so comparisons vary rankings rather than source availability or duplicate-document multiplicity.

The corpus pipeline starts from a frozen USPTO release manifest. It verifies each downloaded file and streams parseable patent archives before identifier-level de-duplication. Seventeen DTD files and one documentation PDF remain as release support files but are not parsed as patent records. Normalization produces one searchable record per publication identifier and retains the more text-complete version when sources collide. Table 7 records release, archive, raw-record, and final-corpus counts.

Historical Full and Current Full are deterministic date slices of one normalized inventory. The snapshot cutoff applies to the collection as a whole; query-specific temporal eligibility requires comparing each candidate's date with its query's date. The corpus and ranking tables measure recorded-citation recovery in Historical Full. Current Full supplies the separate post-2024 manual cases, so later releases cannot alter benchmark candidate membership or metric denominators. PANORAMA gold documents (Lim et al., 2025) preserve only eligible U.S. benchmark targets missing from the official parsed inventory.

Query construction retains only instances whose recorded targets are U.S. publications. Claim-level tasks are keyed by query identifier, section, family identifier, and claim text. We preserve PANORAMA's PAR4PC split assignments: its source protocol groups patent applications by unique identifier before a 90/5/5 allocation (Lim et al., 2025). Different claim contexts and gold sets remain distinct instances within their assigned split. Family membership is retained as metadata, with no additional family-based regrouping. Validation-500 is a deterministic seed-42 hash sample from the official validation split and is disjoint from test. These supervised-data boundaries do not characterize patent content encountered during backbone pretraining. Table 7 provides split and citation counts; Figure 4 summarizes construction.

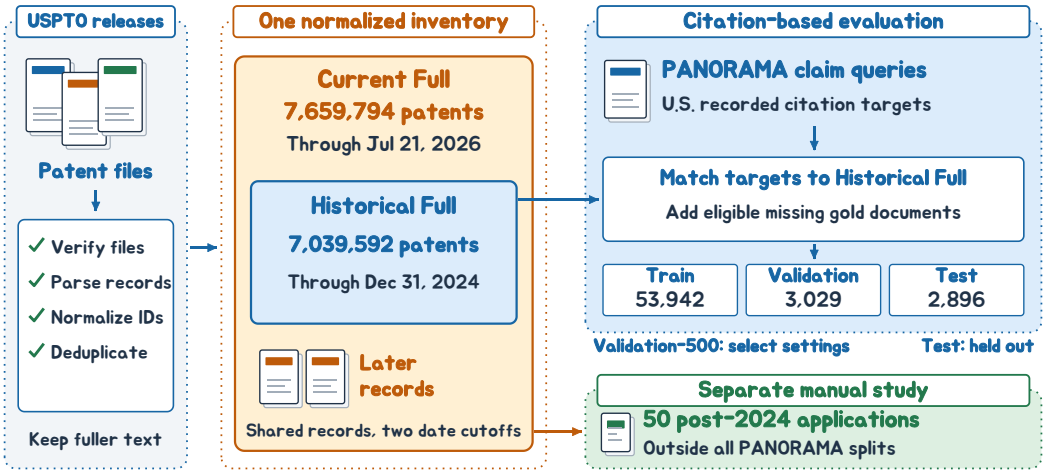

Figure 4: Corpus construction from one normalized inventory. Historical Full supports citation-based evaluation; Current Full supplies the separate manual study. Validation-500 selects configurations, with test queries held out.

Table 7: Reported corpus counts and query-construction rules.

Corpus counts and query construction	
Item	**Count or construction rule**
Raw USPTO input	4,038 official release files, including 4,020 parseable patent archives; 18,713,839 records before de-duplication.
Historical Full	7,039,592 unique U.S. patents published on or before December 31, 2024.
Current Full	7,659,794 unique U.S. patents through July 21, 2026, including 620,202 post-2024 additions.
Benchmark queries	59,953 extracted PAR4PC claim instances; 59,867 remain after requiring U.S.-only recorded targets: 53,942 train, 3,029 validation, and 2,896 test. The retained splits contain 118,336, 6,429, and 6,202 recorded citation labels, respectively.
Identifier collisions	One record per identifier; the more complete text record is retained across sources.
Recorded-gold protection	Every eligible U.S. examiner-cited target is resolved against the frozen inventory or the PANORAMA gold-document supplement before corpus export.
Validation subset	A deterministic seed-42 hash sample of 500 official validation queries; no test query is used for configuration selection.
Family handling	Family membership is retained as metadata and is not itself a de-duplication key.

B Protocols and Reproducibility

B.1 Configuration Selection and Interpretation

Configuration selection uses Validation-500 rather than the test comparison. Table 8 reports crawler, selector, reranker, and candidate-depth results. Backbone sources are Qwen3-Embedding/Reranker (Zhang et al., 2025), Qwen3.5 (Qwen Team, 2026), Llama 3.1 (Grattafiori et al., 2024), and Ministral (Mistral AI Team, 2024).

Table 8: Validation-500 configuration selection (fractions). Bold: panel best excluding $*$ rows; red: selected. $*$: excluded from controlled depth comparisons (Appendix A.2).

(a) Crawler backbone: Qwen3-Embedding					
Size (supervised)	**R@20↑**	**R@100↑**	**R@1000↑**	**R@5000↑**	**MRR↑**
0.6B	0.2717	0.4633	0.7198	0.8412	0.1767
4B	0.3234	0.5234	**0.7611**	**0.8623**	**0.1959**
8B	**0.3239**	**0.5241**	0.7598	**0.8623**	0.1907
(b) Fixed-pool selector and reranker					
Configuration			**R@20↑**	**R@100↑**	**MRR↑**
Crawler order			0.3234	0.5234	0.1925
Qwen3.5-4B LoRA			**0.4658**	0.6879	0.3054
Qwen3.5-9B LoRA			**0.4658**	**0.6883**	**0.3307**
Llama-3.1-8B LoRA			**0.4658**	0.6872	0.3200
Ministral-8B LoRA			0.3972	0.6210	0.2968
Qwen3-Reranker-8B zero-shot			0.3040	0.4671	0.2512
Qwen3-Reranker-8B LoRA			0.3441	0.5522	0.2863
(c) Candidate-pool depth					
Input candidates			**R@20↑**	**R@100↑**	**MRR↑**
100*			0.4427	0.6273	0.3071
500			**0.4658**	0.6651	0.3215
1000			**0.4658**	**0.6883**	**0.3307**

The crawler is selected by Recall@1000, with Recall@5000 and MRR reserved for ties. Selector backbones are selected by Recall@20, then MRR and Recall@100. The 4B, 9B, and Llama selectors share validation Recall@20 of 0.4658; MRR selects 9B (0.3307 versus 0.3054 and 0.3200). This choice precedes the test comparison. Depth selection uses Recall@20, then Recall@100 and MRR.

Between 500 and 1,000 candidates, equal Recall@20 and higher Recall@100 favor 1,000. The starred 100-input row remains separate from this controlled depth comparison.

Table 2 reports citation-based model comparisons; Table 6 reports the separate human evaluation. Table 3(a) reports the negative-source ablation. The effects are not additive; they show the contribution of both sources under the tested setup but do not provide an independent causal decomposition.

B.2 Training Targets and Negative Semantics

The crawler objective is defined over claim–citation pairs. Each term assigns its positive role to one recorded target, while the full set $\mathcal{G}_i$ supplies evaluation targets and exclusions for explicit BM25 mining. The denominator includes the designated target and gathered batch positives; the numerator contains only the designated target. Explicit BM25 exclusion therefore differs from the pair-level competition among batch documents. A document's positive role for one training pair does not establish its negative status for another, and citation-negative examples can still be technically relevant. Removing BM25 negatives and cross-device gathering leaves local batch comparisons active. Table 3(a) evaluates these sources under the same single-positive objective.

The selector uses response-only language-model training. The 120,000 examples count claim–patent pairs, so multiple examples can share a claim. The released preparation scripts take train or validation queries as separate inputs, match retrieved lists to their query keys, retain recorded targets as positives, and mine retrieved candidates outside each query's target set as negatives. This split-specific construction distinguishes supervision pairs from unique claims. Prompt-token labels are masked, and target JSON contains a citation decision, a fixed label-associated score, and a reason: `gold examiner citation` for positives or `hard negative candidate` otherwise. These strings serialize labels; ranking uses the decoded label and score. The two yes/no controls share one model and vary likelihood sorting versus a positive-first gate followed by crawler order; the structured selector is separately trained.

B.3 Implementation Details

Table 9 specifies optimization, software, inference, and API boundaries. Dense-only is the reported dense-retrieval baseline, evaluated using its native ranking. Its checkpoint is not specified in the configuration summary, so it is not identified with any named Qwen3 checkpoint. The named before/after Qwen3-Embedding pairs provide the same-backbone adaptation controls. Crawler adaptation uses LoRA (Hu et al., 2022); quantized selector adaptation uses QLoRA (Dettmers et al., 2023).

Table 9: Implementation settings and evaluation interfaces.

Implementation and interface settings	
Component	**Setting**
Compute and software	The reported 9B and 8B selector training runs and timed local-selector inference used one Linux node with eight 32 GB RTX 5090 GPUs. Software versions were Python 3.12.13, PyTorch 2.12.0+cu130, Transformers 5.13.0, Accelerate 1.14.0, PEFT 0.19.1, and bitsandbytes 0.49.2.
Crawler inputs and loss	Maximum sequence length 512; last-token pooling and ℓ_2 normalization; contrastive loss with temperature 0.03, cross-device negatives, and eight explicit BM25 hard negatives per query.
Crawler adaptation	Qwen3-Embedding-4B with LoRA rank 16, scale 32, dropout 0.05, and target modules `q_proj`, `k_proj`, `v_proj`, and `o_proj`. Optimization used AdamW, learning rate 2×10^{-5}, weight decay 0.01, one epoch, 3% warm-up, a cosine schedule, BF16, and seeds 42, 43, and 44. Eight distributed processes used per-device batch 4 and gradient accumulation 2 (effective global batch 64).
Selector training data	120,000 structured claim–patent examples comprising 25,626 recorded-citation positives and 94,374 retrieved citation-negative examples. Positive targets use `true`/0.95, and citation-negative targets use `false`/0.05. These scores are ordering labels rather than calibrated probabilities.

Continued on next page

Table 9 (continued)

Component	Setting
Reported selector adaptation	The language-only Qwen3.5-9B load uses 4-bit NF4 QLoRA, double quantization, BF16 compute, and maximum sequence length 3072; no visual input or visual parameter is used. LoRA used rank 16, scale 32, dropout 0.05, and all query, key, value, output, gate, up, and down projection modules. Training used one epoch, learning rate 1.5×10^{-4}, per-device batch 2, gradient accumulation 4 (effective global batch 64), gradient checkpointing, seeds 42, 43, and 44, and eight distributed RTX 5090 processes.
Selector backbone controls	Llama-3.1-8B and Ministral-8B use the same 120,000 examples, target format, maximum sequence length 3072, QLoRA settings, epoch count, and learning rate as the 9B pipeline. Per-device batch 2 and accumulation 4 preserve global batch 64. The reported Qwen3.5-4B comparator used maximum sequence length 2048, learning rate 2×10^{-4}, per-device batch 1, and accumulation 8. We therefore do not interpret it as a strict scaling ablation.
Selector inference	Each top-1000 crawler candidate is judged independently from a bounded claim-conditioned patent context. Candidates are ranked by predicted label and generated score. Greedy decoding uses at most 96 new tokens, 4-bit NF4 loading, and a top-100 output. All local backbones use eight concurrent shards, one process per RTX 5090 GPU, and inference batch 16.
Fixed-pool API selector	The comparison uses the same top-1000 pool and top-100 output. Returned IDs are restricted to the input pool and de-duplicated; omitted slots are completed in crawler order. Appendix B.4 documents batched pointwise and listwise interfaces. The configuration summary does not identify the interface or batch size for each reported API row.
Raw-claim API full flow	Each tested configuration jointly performs retrieval and final selection from the claim. Open output allows an empty final list; forced output requires a nonempty list. Appendix B.5 defines the comparison boundary. These rows evaluate complete search configurations, separately from fixed-pool rerankers.

The crawler uses temperature 0.03, cross-device batch comparisons, and eight explicit BM25 negatives per query (Robertson & Zaragoza, 2009). Its 512-token patent representation concatenates title, abstract, and claims. The selector instead constructs a claim-conditioned excerpt from the full available fields. These are distinct stage inputs: same-encoder adaptation comparisons assess learning within the crawler representation, while fixed-pool ranking evaluates the supplied candidate texts and ranking interfaces. Alternative long-document representations and hybrid systems provide complementary design choices to the tested single-vector crawler.

B.4 Prompt, Serialization, and Output Schema

The following templates specify local-selector, fixed-pool, and raw-claim input and output interfaces. Bracketed fields denote serialized inputs. The API templates describe the message format; Appendix B.5 explains the scope of the complete search configurations.

Local selector.

The system message is:

```
You are a patent citation selector. Decide whether
the candidate patent is likely to match a recorded
examiner citation for the target claim. Use only the
supplied text. Return JSON only.
```

The user message is:

```
Target claim: [CLAIM]
Candidate patent ID: [PATENT_ID]
Candidate text: [CANDIDATE_TEXT]
Return JSON: {"is_prior_art": true/false, "score":
0.0--1.0, "reason": "..."}
```

The required schema is therefore

```
{"is_prior_art": true/false, "score": 0.0--1.0,
"reason": "..."}.
```

Malformed JSON, a missing Boolean decision, or a score outside $[0, 1]$ is invalid rather than assigned a fabricated score. Valid candidates are ordered by predicted label and score, with crawler order resolving exact ties; unresolved candidates remain in crawler order after valid decisions.

Fixed-pool API selector.

Every batch begins with:

```
You are evaluating candidate patents against a target
patent claim.  Use concrete technical limitations
rather than topic similarity.  Do not invent IDs.
```

For the batched-pointwise interface, this is followed by:

```
Judge every candidate independently.  Return one item
for every supplied candidate, using only supplied
IDs, as strict JSON:
{"decisions":[
{"patent_id":"...","is_prior_art":true,
"score":0.0,"reason":"..."}]}
```

For the batch-listwise interface, it is followed instead by:

```
Rank every supplied candidate.  Return every supplied
ID exactly once as strict JSON:
{"ranked_patent_ids":["..."]}
```

Both interfaces then append `Target claim: [CLAIM]` and numbered blocks of the form `[INDEX] patent_id=[PATENT_ID]` followed by `[CANDIDATE_TEXT]`. Parsing canonicalizes identifiers, rejects out-of-pool IDs, and removes duplicates. Remaining output slots are completed in crawler order. These templates document the available interfaces; the configuration summary in Table 9 distinguishes them from row-specific execution settings.

Raw-claim full flow.

The raw-claim prompt is:

```
Given the patent claim below, return real patent
publication or grant IDs that are technically
relevant.  Do not invent IDs.  Return JSON only as
{"patent_ids":["..."]} with at most [MAX_RESULTS] IDs.
Claim:  [CLAIM]
```

The open policy preserves an empty parsed list; forced output imposes a nonempty-output condition on the complete configuration. The prompt above specifies the requested response format, not the search-tool configuration or the implementation of the forced-output condition.

B.5 Raw-Claim API Protocol and Interpretation

The complete open and forced-nonempty results appear in Table 5. They are distinct from fixed-pool API selection: a nonempty final list is not the same constraint as returning exactly 100 supplied candidates. The manual evaluation uses forced full-flow configurations, whereas the open rows diagnose empty-output behavior.

The open and forced rows are evaluated as complete configurations because their retrieved lists also differ. Final Recall@20 measures citation recovery after each configuration's final selection, including zero credit for an empty list. The observed change therefore combines retrieval and output behavior; it is not a controlled estimate of the effect of filling an empty result. Empty counts describe the final output and do not by themselves distinguish abstention from request or parsing failures. Reproducing the configurations requires their model, prompt, search tools, decoding settings, and output handling together; the message template alone is insufficient.

Table 5 reports the tested API configurations. Browser-based products can use different search tools, prompts, candidate sources, and interaction budgets, so a model name alone does not identify a comparable search system. The method conclusions rest on the same-encoder adaptation and fixed-pool ranking comparisons; the raw-claim rows provide complementary observations of retrieval and output behavior.

B.6 Context Construction and Adaptation Details

Section 4.3 defines the selector's fixed input and output interface. Field priorities favor title, abstract, and claim passages when term-overlap scores are comparable. The 1,200-word budget bounds passage assembly before token truncation. The target scores 0.95 and 0.05 encode positive and negative citation labels; ranking uses the predicted label first and score second. These settings specify the tested configuration.

Table 9 gives the separate crawler and selector training settings. Selector serialization preserves the full target response and truncates the prompt to the sequence limit, retaining label and score supervision.

For selector depth m, inference batches and shards m independent claim–patent evaluations as specified in Table 9. The 3,072-token sequence limit is an input ceiling; actual input lengths depend on the claim and excerpt. Moving from 500 to 1,000 candidates doubles pair evaluations, while Table 10 reports the associated recall changes. Candidate evaluations and Table 4's estimated downstream inspection tokens measure separate budgets. End-to-end latency and GPU cost additionally depend on execution time and hardware utilization. Dynamic depth allocation has also been studied with 64-dimensional patent embeddings (Onozuka & Ohnishi, 2026); here, Validation-500 selects a shared depth before test evaluation.

The fallback ensures a defined output even when a generated response cannot be parsed. Predicted citation-negative candidates remain valid ranking entries, whereas malformed responses follow all valid decisions. This distinction matters because selecting by the yes/no label alone would discard documents, while the implemented policy orders the entire candidate pool. Algorithm 1 in Appendix B.8 specifies this interface independently of any optional inspection rule.

B.7 Inspection Coverage Definition

To study how much of the ranked list is inspected before additional patents cease to cover new claim elements, we analyze a fixed prefix using an element-coverage proxy. Let $\mathcal{E}_i$ be elements obtained from the claim and $b_i(e, d) \in \{0, 1\}$ the proxy decision for element e and patent d. Coverage after inspecting prefix D_t is

$$\text{Coverage}_i(t) = \frac{1}{|\mathcal{E}_i|} \sum_{e \in \mathcal{E}_i} \mathbf{1}[\exists d \in D_t : b_i(e, d) = 1]. \tag{6}$$

The proxy specification combines content-token overlap (scaled by 1.6, capped at 0.95, and thresholded at 0.72), a full-support indicator, and a patent date no later than the query application date. The binary decision $b_i(e, d)$ represents their conjunction. These are fixed analysis settings, shared across patience values. Element-level support indicators are inputs to this analysis, distinct from the selector's patent-level citation labels. The date condition applies to inspection support, separately from the corpus-recovery metric.

A patience policy tracks consecutive patents that add no newly covered elements and resets the count whenever coverage increases. It stops at complete coverage, after p no-gain patents, or at the inspection cap of 20 (Algorithm 2). Thus p is not a minimum prefix length: the complete-coverage exit permits fewer than p inspections. Table 4 compares policies using the same ranking and fixed support decisions.

The anonymous release provides architecture code, including a separate lexical coverage diagnostic. Reproducing the fixed-proxy sweep additionally requires its element-support decisions and token accounting; the diagnostic is not the implementation that generated Table 4. Proxy coverage and estimated token savings characterize this inspection analysis, rather than measured reviewer time or end-to-end retrieval latency.

B.8 Inference and Stopping Policy

Algorithm 1 produces the ranked output; the inspection policy then traverses its top-20 prefix.

Algorithm 1 Retrieval and pointwise ranking in ClaimCrawl

Input: claim q, frozen corpus index, trained crawler ϕ, trained selector θ
Retrieve $\mathcal{C}^{5000}$ in descending inner-product similarity to q
Let $\mathcal{C}^{1000}$ be its leading 1,000 candidates; initialize valid list A and fallback list F
for each d in $\mathcal{C}^{1000}$, retaining its crawler rank r_d **do**
    Build the bounded claim-conditioned excerpt $x(q, d)$
    Decode and parse a structured selector response
    if the response contains a valid decision and score $(\hat{y}_d, s_d)$ **then**
        Append $(d, \hat{y}_d, s_d, r_d)$ to A
    else
        Append (d, r_d) to F
    end if
end for
Sort A by descending $(\hat{y}_d, s_d)$ and ascending r_d for ties
Append F in crawler order and return the leading 100 patents

The controller updates coverage after each patent without reordering documents. On the test set, $p = 7$ and $p = 10$ retain 94.3% and 96.4% of all-20 proxy coverage, with estimated token savings of 61.8% and 52.5%. Both have an unsupported-element rate of 0.0114. Increasing patience therefore gains coverage while using more inspection tokens under this proxy.

Policy selection uses Validation-500. Its all-top-20 reference has coverage 0.7783 and Unsupported 0.0092. Patience values 2, 3, 5, 7, and 10 yield Unsupported values of 0.0519, 0.0276, 0.0144, 0.0022, and 0.0022. The selection rule first excludes policies whose rate exceeds the reference, leaving 7 and 10, then maximizes estimated token savings. It selects $p = 7$ (58.65%) over $p = 10$ (49.65%). This validation criterion fixes the test operating point; maximizing test coverage alone would favor a larger inspection budget.

Algorithm 2 Patience-based inspection with fixed proxy decisions

Input: ranked patents R_i, claim elements $\mathcal{E}_i$, fixed proxy $b_i(e, d)$, inspection cap $L = 20$, patience p
Initialize covered set $S = \varnothing$ and no-gain count $c = 0$
for each patent d in the first L entries of R_i **do**
    $S' \leftarrow S \cup \{e \in \mathcal{E}_i : b_i(e, d) = 1\}$
    if $S' = S$ **then**
        $c \leftarrow c + 1$
    else
        $c \leftarrow 0$
    end if
    $S \leftarrow S'$
    if $S = \mathcal{E}_i$ or $c \geq p$ **then**
        Stop inspection
    end if
end for
Output: inspected prefix and covered elements S

C Additional Results and Diagnostics

This section reports additional recall cutoffs, target-set completeness, training-seed variability, and inspection-proxy diagnostics. Raw-claim API scores appear only in Table 5; their protocol is discussed in Appendix B.5.

C.1 Candidate Depth and Ranking Controls

Table 10(a) reports candidate-depth results with 100 output patents at each input depth. Increasing the input from 500 to 1,000 leaves Recall@20 at 51.27% but raises Recall@100 from 70.54% to 73.32%. All reported 100-input metrics are retained, with $*$ distinguishing that configuration from the controlled 500-versus-1,000 comparison (Appendix A.2).

Panels (b,c) supplement the main comparison with additional ranking and completeness metrics. Binary log-odds sorting uses the relative likelihood of the yes/no citation labels; the gate control uses the same trained model followed by crawler order. These controls differ from the separately trained reranker and unadapted selector in the main table. Equal values at one cutoff do not identify equal systems.

Table 10: Candidate-depth and additional test metrics (fractions; 100 outputs). Bold: panel best excluding $*$ rows; red: selected. $*$: excluded from controlled depth comparisons (Appendix A.2).

(a) Candidate-pool depth

Input depth	R@10↑	R@20↑	R@50↑	R@100↑	All-Gold@100↑	MRR↑
100*	0.4333	0.4988	0.5058	0.5058	0.3117	0.3391
500	**0.4333**	**0.5127**	**0.6364**	0.7054	0.5608	0.3434
1000	**0.4333**	**0.5127**	**0.6364**	**0.7332**	**0.5817**	**0.3438**

(b) Selector controls: additional recall cutoffs

Configuration	R@10↑	R@50↑
Yes/no: label-likelihood ranking	0.1545	0.3472
Yes/no: gate + crawler order	**0.3194**	**0.4479**

(c) Fixed-pool target-set completeness

Configuration	All-Gold@100↑
Crawler order	0.3441
Qwen3.5-9B no LoRA	0.1650
Qwen3-Reranker-8B zero-shot	0.4258
Qwen3-Reranker-8B LoRA	0.4783
Yes/no: label-likelihood ranking	0.4167
Yes/no: gate + crawler order	0.4167
ClaimCrawl selector 9B	**0.5817**

C.2 Interpretation of Additional Metrics

The two yes/no controls have equal All-Gold@100 but different early recall and MRR, distinguishing recovery of complete target sets from the order in which targets appear.

Coverage retention is $100C_p/C_{20}$ and inspection savings are $100(1 - I_p/20)$, where C_p is mean proxy coverage and I_p is mean inspected patents. Table 4 supplies these inputs. Estimated token savings use the recorded token-cost proxy and cannot be inferred from document counts alone.

Unsupported-element rate. On the 2,896-query test, Unsupported is 0.0743, 0.0457, 0.0286, 0.0114, and 0.0114 for $p = 2, 3, 5, 7, 10$, with 0.0114 for all 20. Both 7 and 10 match the all-20 reference on this diagnostic while reducing inspection. These are test results for the fixed policies; selection uses the separate validation values in Appendix B.8.

C.3 Training Variability

Table 11 reports variation across the three training seeds.

These standard deviations measure variation among the reported trained runs. They are not confidence intervals for a difference between methods, which requires aligned query-level outcomes from both systems.

Table 11: Three-seed test results. Recall: R@5000 for crawler, R@20 for selector. Bold rows report mean and standard deviation.

Stage	Seed	Recall↑	MRR↑
(a) Crawler: three training seeds			
Crawler	42	0.9365 R@5000	0.2136
Crawler	43	0.9288 R@5000	0.2098
Crawler	44	0.9347 R@5000	0.2111
Crawler	**Mean±SD**	$\mathbf{0.9333 \pm 0.0040}$	$\mathbf{0.2115 \pm 0.0019}$
(b) Selector: three training seeds			
Selector	42	0.5127 R@20	0.3438
Selector	43	0.4998 R@20	0.3421
Selector	44	0.5116 R@20	0.3362
Selector	**Mean±SD**	$\mathbf{0.5080 \pm 0.0072}$	$\mathbf{0.3407 \pm 0.0040}$

D Manual Evaluation and Failure Handling

D.1 Output Integrity and Failure Handling

The fixed-pool protocol requires checks of query count, candidate membership, duplicate identifiers, output length, and deterministic tie handling. Selector candidates must belong to the supplied crawler prefix. For fixed-pool APIs, invalid or repeated identifiers are removed and unfilled positions are completed from crawler order. These rules preserve a 100-patent output without adding out-of-pool documents. Appendix A.2 specifies the comparison boundary for the 100-input configuration.

Raw-claim full flow outputs use a different boundary. The open policy retains an empty selected list; forced output requires a nonempty final list. Retrieved and selected lists are evaluated separately, and the output-policy comparison concerns complete configurations (Appendix B.5). In the manual study, missing, duplicate, self-matching, or unresolved items retain their returned positions and count as rejected. Post hoc cleaning therefore cannot improve a method's top-20 result.

D.2 Stage-Specific Failure Diagnostics

A stage-specific diagnostic requires the per-query candidate and output lists. For each recorded citation, membership in the crawler top 5,000, handed-off top 1,000, selector top 100, and inspection prefix distinguishes where it ceases to be available. Raw-claim search instead compares retrieved and selected identifiers. Empty output is a separate query-level condition.

These procedures define how to localize losses; the reported aggregate metrics are not an additive stage-by-stage loss decomposition. Later-stage losses are conditional on earlier availability and cannot be summed as independent causes. Per-query identifiers, targets, ranks, unresolved identifiers, and duplicate removals provide the information needed to trace each classification.

D.3 Manual Evaluation Protocol

The 50 cases are post-2024 USPTO applications selected from Current Full and excluded from every PANORAMA split. They have no examiner-citation labels for this evaluation. Before review, results from all methods are pooled and de-duplicated by query and patent identifier. Figure 5 summarizes the review and aggregation workflow.

Review form and blinding. Each displayed item contains an anonymous item identifier, target claim, candidate patent identifier, title, abstract, and claim-conditioned excerpt. Display order is randomized; method identity and returned rank are hidden. Each reviewer records one binary decision: accept the patent if it is relevant to the query or provides substantial technical assistance for investigating it; otherwise reject it. Broad topical overlap alone is insufficient.

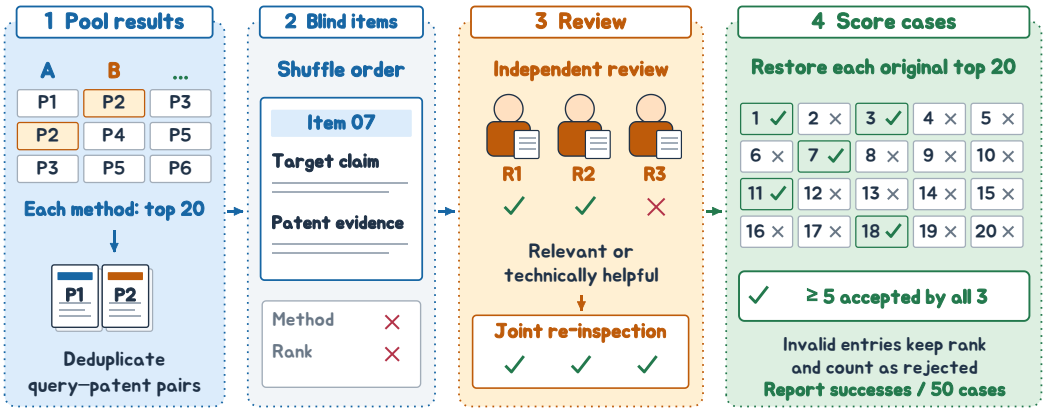

Figure 5: Blinded review of 50 later applications. Three reviewers independently assess items and jointly resolve disagreements. Success requires at least five unanimously accepted patents in the original top 20. Displayed items and judgments are schematic.

Independent review and adjudication. Three reviewers assess each pooled item independently before joint re-inspection of disagreements. A patent contributes to a method when all three accept the final decision. Reported case counts use this adjudicated consensus. Agreement statistics would instead use the initial independent labels, so unanimity after discussion is not an inter-rater agreement estimate.

Case aggregation and failures. A case succeeds when at least five patents in a method's top 20 satisfy the unanimous criterion. This endpoint asks whether an early list supplies several useful patents rather than reproduces a sparse citation target. Missing, duplicate, self-matching, or unresolved outputs retain their returned positions and are rejected rather than silently removed. The main paper reports method-level case counts; this supplement defines their construction.

Scope of manual judgments. The 50 cases provide a temporal extension, not another citation-recall estimate. Rejected candidates may still have technical value. Legal assessment requires complete disclosures, applicable dates, and broader prior art. Constructed citation mismatches illustrate label boundaries, not observed system failures.

D.4 Uncertainty of Case-Level Success

We summarize uncertainty in Table 6 using two-sided 95% Wilson intervals (Wilson, 1927). For k successful cases out of $n = 50$, let $\hat{p} = k/n$ and $z = 1.96$. The interval is

$$\frac{\hat{p} + z^2/(2n) \;\pm\; z\sqrt{\hat{p}(1-\hat{p})/n + z^2/(4n^2)}}{1 + z^2/n}. \tag{7}$$

Applied to the reported counts, the intervals are 12.8–35.2% for BM25, 15.9–39.6% for BGE-M3, 19.1–43.8% for Dense-only, 22.4–47.8% for DeepSeek V4-Pro forced, 31.2–57.7% for GPT-5.5 forced, and 44.2–70.6% for ClaimCrawl. These are calculations from the existing success counts, not additional trials.

The intervals assume independent cases with a common success probability; they do not account for case-selection bias or correlated applications. Methods share the same cases, so a paired comparison requires each method's success indicator on each case. Marginal counts cannot recover those joint outcomes. We therefore report descriptive case-level differences without a paired significance claim.