RecruitBench: An Outcome-Grounded Benchmark for Evaluating AI Recruiting Systems

Aditya Sood
Department of Computer Science
Stanford University
adibsood@stanford.edu

Advisor: Prof. Mykel J. Kochenderfer

Abstract

AI recruiting systems are increasingly used to screen and rank candidates, yet existing evaluations rely on proxies like keyword overlap, engagement, and human annotation labels. We present **RecruitBench**, an outcome-grounded benchmark built from recruiter-submitted candidates paired with observed interview progression. RecruitBench contains 1,594 resume–job pairs across 18 startup roles and captures a real setting in which candidates have already been screened by professional recruiters, making discrimination substantially harder than open-pool resume filtering. The benchmark defines two real-world tasks: interview advancement prediction for resume–job pairs and resume-to-K job recommendation. RecruitBench also includes a synthetic stress-test suite that isolates robustness failures like adversarial keyword manipulation. Across both tasks, classical lexical baselines perform near chance, while LLM-based systems show substantially stronger discrimination. The best screening method reached an AUC-ROC of 0.685 in predicting interview progressions. For task 2, our benchmark indicated that a parallel batched ranking architecture improves recall, calibration, and latency over monolithic prompting. RecruitBench offers a realistic benchmark for evaluating whether AI recruiting systems are predictive and robust in the workflow settings they are designed to support.

1 Introduction

Algorithmic decision support is now deeply embedded in the recruitment pipeline, from talent sourcing and search to resume screening and candidate ranking. Professional standards in recruiting and U.S. enforcement guidance increasingly treat such tools as employment "selection procedures" when they meaningfully influence hiring decisions. This classification raises the bar for validation, monitoring, and documented evidence that model scores support their intended use. Yet, contemporary recruiting systems face a measurement problem: how can we evaluate whether automated screening and ranking models are predictive of real hiring progression, rather than merely correlating with superficial text similarity? [U.S. Equal Employment Opportunity Commission(2023)]

This paper introduces RecruitBench, a benchmark built from recruiter-sourced candidate submissions paired with observed hiring progressions. Prior work on resume–job matching typically relies on proxy labels such as engagement signals or curated match annotations rather than observed hiring outcomes [Maheshwary et al.(2018)Maheshwary, Anand, et al., Bian et al.(2020)Bian, Li, et al., Kenthapadi et al.(2017), Organizers(2016)]. In RecruitBench, professional recruiters proactively identify candidates and submit them to hiring teams. The first challenge is not filtering out obviously irrelevant applicants, but discriminating among candidates that are all plausibly relevant.

Thus, we arrive at our first formalized task: interview advancement prediction. Given a resume–job pair, predict whether the candidate reaches a first-round interview.

The second challenge is ensuring the right candidates are matched with the right jobs to improve the efficiency of recruiting marketplaces. Modern recruiting platforms rarely evaluate a candidate against a single job in isolation. Instead, recruiters typically begin with a candidate and search for roles that best match that candidate's experience. In production systems, this is implemented as a retrieval-and-ranking pipeline: a retrieval layer first identifies a candidate-specific pool of potentially relevant jobs, after which a scoring model ranks these jobs and surfaces the top matches to recruiters. The effectiveness of such systems therefore depends not only on screening accuracy for individual resume–job pairs, but also on the ability to correctly prioritize the best opportunities for a candidate among many plausible roles. Resume-to-K job recommendation is the second task, where the system must identify strong roles for a resume.

RecruitBench contains 1,594 recruiter-submitted candidates spanning 18 high-volume roles. Pass rates vary from 11.8% to 52.7% across jobs, making the benchmark difficult to solve using global thresholds. (e.g., always predict negative).

Outcome logs alone may not isolate specific failure modes of scoring systems. For example, production data may not reveal whether a model over-rewards keyword stuffing, is unduly sensitive to formatting artifacts, or properly penalizes domain mismatch. RecruitBench therefore augments observed data with controlled synthetic resumes that vary along interpretable dimensions such as experience alignment, company pedigree, and adversarial manipulation. These synthetic cases are associated with predefined expected score ranges, enabling clear robustness evaluation that complements aggregate predictive performance. [Ribeiro et al.(2020)Ribeiro, Wu, Guestrin, and Singh, Liu and de Rijke(2024)]

RecruitBench contributes (i) an outcome-grounded benchmark tied to observed interview progression among recruiter-curated candidates, (ii) task formulations aligned with operational recruiting scenarios in screening and recommendation, and (iii) a structured methodology for robustness testing via controlled synthetic stress cases. By grounding evaluation in real outcomes, the benchmark enables direct measurement of whether AI recruiting systems are predictive in the decision setting they are designed to support.

2 Related Work

RecruitBench lies at the intersection of information retrieval, learning to rank, reciprocal recommendation, and algorithmic hiring.

2.1 Job–resume matching and talent search

Substantial studies analyze job–resume matching by learning semantic compatibility between resumes and job descriptions. [Maheshwary et al.(2018)Maheshwary, Anand, et al., Bian et al.(2020)Bian, Li, et al., Yu et al.(2024)] Industrial recruiting platforms have also described large-scale talent search and recommendation systems, emphasizing multi-stage retrieval, ranking, and personalization. These systems are closely related to RecruitBench in architecture, but they typically optimize for platform objectives such as recruiter engagement and response rather than interview progression. [Geyik et al.(2018)Geyik, Kenthapadi, et al.]

RecruitBench differs from prior job–resume benchmarks by grounding evaluation in candidate advancement to a first-round interview within a recruiter-mediated submission pipeline. Our benchmark is explicitly downstream of initial relevance filtering.

2.2 Information retrieval and learning to rank

RecruitBench's resume-to-job recommendation task is aligned with information retrieval (IR) and learning-to-rank strategies. Lexical algorithms like BM25 (keyword matching) remain strong baselines for these tasks. [Robertson and Zaragoza(2009), Manning et al.(2008)Manning, Raghavan, and Schütze]

Whereas traditional IR benchmarks primarily measure general relevance, RecruitBench evaluates ranking quality against a downstream outcome: whether candidates advance to interview, conditioned on recruiter selection. This shifts evaluation from annotator-defined to outcome-grounded.

2.3 Job recommendation

Job recommendation can be viewed as a form of reciprocal recommendation in which recommendations must satisfy constraints and incentives of more than one party. Public benchmarks often use interaction data-clicks, applications, impressions, or expressed interest—to train and evaluate models. [Organizers(2016), Kenthapadi et al.(2017), de Ruijt et al.(2021)]

RecruitBench's resume-to-K task is structurally similar to job recommendation, but differs in supervision: it uses a high-stakes progression label (interview advancement) rather than engagement. This distinction is important because engagement may reflect exposure effects, interface behavior, or job-seeker browsing preferences, whereas interview progression more directly reflects employer-side screening outcomes.

2.4 Robustness and fairness in algorithmic hiring

Strong average performance does not guarantee that a model behaves reliably in all cases. A system may achieve high overall accuracy while still failing in predictable ways on specific types of inputs. This has motivated work on robustness evaluation, including behavioral testing frameworks in NLP that function like unit tests: they check whether models responded appropriately to controlled perturbations [Ribeiro et al.(2020)Ribeiro, Wu, Guestrin, and Singh]. Similar concerns have led to increased attention to robustness in information retrieval, where researchers examine how ranking systems behave under atypical queries [Liu and de Rijke(2024)].

In parallel, research on algorithmic hiring highlights risks related to fairness and transparency. Studies show that choices about training data and prediction targets can encode or amplify existing inequalities [Raghavan et al.(2020)Raghavan, Barocas, Kleinberg, and Levy, Barocas and Selbst(2016)]. U.S. regulatory guidance increasingly treats automated screening tools as employment selection procedures. Recent local regulation, such as New York City's Local Law 144, further requires bias audits and public disclosures for certain automated employment decision tools [U.S. Equal Employment Opportunity
New York City Department of Consumer and Worker Protection(2023)].

RecruitBench does not assert that observed interview decisions are optimal or unbiased. Instead, it provides measurement infrastructure for (i) predictive alignment with observed outcomes in a realistic workflow setting and (ii) controlled stress testing to diagnose model sensitivities that outcome logs alone may not reveal.

3 RecruitBench Benchmark

RecruitBench is constructed from recruiter-sourced candidate submissions paired with observed hiring outcomes. The dataset captures the progression of candidates through a structured hiring pipeline and provides outcome labels derived from applicant tracking system (ATS) records. In addition to production data, RecruitBench includes a set of controlled synthetic resumes designed to evaluate model robustness under specific edge cases.

3.1 Data Source and Collection

The production portion of RecruitBench contains **1,594 recruiter-submitted candidates across 18 startup job openings**. Candidates were proactively sourced by professional recruiters and submitted to hiring teams for evaluation.

RecruitBench therefore does *not* represent an open applicant pool. Candidates were not self-selected applicants responding to job postings. Instead, each candidate was identified and screened by a recruiter before being presented to the hiring organization. As a result, most candidates already satisfy baseline relevance criteria for the role, making the evaluation task substantially more difficult than traditional resume filtering.

Each observation in the dataset corresponds to a **resume–job pair** and includes:

- the candidate resume text
- the associated job description (green flags, red flags, requirements)
- transcript with the hiring manager discussing what they want in their candidates
- the candidate's observed progression through the hiring pipeline

Resumes were anonymized prior to inclusion in the dataset. Personally identifiable information such as names, contact details, and other direct identifiers were removed.

3.2 Outcome Labels

Candidate outcomes are derived from the hiring pipeline recorded in the ATS used by participating companies. After recruiters submit candidates, hiring teams evaluate them through several stages before scheduling interviews.

RecruitBench uses the following simplified funnel representation:

1. **Administrative Screening** — hiring team evaluation (first pass).
2. **Company Resume Review** — hiring team evaluation (second pass).
3. **Interview Stages** — candidates who pass resume review are invited to one or more interview rounds.
4. **Late-Stage Outcomes** — candidates may progress to onsite interviews, offers, or hiring decisions.

This benchmark focuses on whether a candidate reaches an interview stage. This event reflects an explicit decision by the hiring organization to invest time in evaluating the candidate further.

A binary label is defined as:

$$y = \begin{cases} 1 & \text{if the candidate reaches an interview stage (pipeline stage } \geq 3) \\ 0 & \text{otherwise} \end{cases}$$

Across the full dataset, the label distribution is:

Label	Count	Percentage
Interview (positive)	496	31.1%
No interview (negative)	1,098	68.9%
Total	1,594	100%

Table 1: Binary interview advancement labels in RecruitBench.

Although all candidates were sourced and screened by recruiters, only about one-third ultimately progressed to interview stages. That said, interview advancement rates varied widely:

Statistic	Value
Mean	29.9%
Standard deviation	11.3%
Minimum	11.8%
Maximum	52.7%

Table 2: Variation in interview pass rates across jobs.

The large spread in pass rates means that a model calibrated to the global average may perform poorly on individual jobs. Effective screening systems must therefore adapt to job-specific decision boundaries.

3.3 Data Splits

RecruitBench defines a stratified **70/30 train–test split** by submission identifier within each job. This ensures that every job appears in both splits while preserving the original class balance.

Split	Candidates	Jobs	Positive Rate
Train	1,124	18	31.5%
Test	470	18	30.2%

Table 3: Train–test partition of RecruitBench.

The train set is primarily used to construct retrieval indices for retrieval-augmented screening systems. It was also used to tune the thresholds and other hyper-parameters for different approaches.

3.4 Synthetic Stress-Test Cases

Outcome data alone does not reveal how models behave under unusual or adversarial inputs. To complement the production dataset, RecruitBench includes a **synthetic stress-test suite of 26 resumes** designed to isolate specific scoring behaviors.

Synthetic resumes are generated using a large language model conditioned on real job descriptions and structured prompts. Each resume is paired with a job and constructed to represent a specific scenario.

The synthetic set includes three categories:

- **Controlled variants (8 cases)** — systematic manipulations of candidate quality, such as perfect matches, missing hard requirements, or overqualification.
- **Edge cases (11 cases)** — realistic but ambiguous profiles including career changers, founders, candidates with career gaps, and regional mismatches.
- **Adversarial cases (7 cases)** — resumes designed to exploit weaknesses in keyword-based systems, including keyword stuffing, inflated metrics, and irrelevant prestige signals.

Each synthetic resume is paired with a real job and assigned an expected score range based on predefined business logic (e.g., perfect match: [8.5, 10.0], hard requirement miss: [2.5, 4.5]). These cases enable evaluation of scoring calibration and behavioral consistency beyond predictive accuracy.

4 Tasks and Evaluation

RecruitBench evaluates recruiting systems in two production-aligned decision settings: (i) *screening*, where a system scores a single resume–job pair, and (ii) *recommendation*, where a system ranks a small pool of jobs for a given resume. Both tasks use the same outcome-grounded label derived from the ATS hiring funnel (Section 3.2). For all LLM-based experiments, the underlying model is fixed: gpt-5.2-2025-12-11.

4.1 Task 1: Interview Advancement Prediction

Problem. Let $\mathcal{R}$ be the set of resumes and $\mathcal{J}$ the set of job postings. Each example is a pair $(r, j) \in \mathcal{R} \times \mathcal{J}$ with label $y_{r,j} \in \{0, 1\}$, where $y_{r,j} = 1$ indicates that the candidate reached an interview stage in the pipeline (pipeline stage ≥ 3 in Section 3.2). A system outputs either a probability $\hat{p}_{r,j} \in [0, 1]$ or a continuous score $s_{r,j}$; when systems naturally output scores on a 0–10 scale, we report results in that scale.

Metrics. We report AUC-ROC as the primary discrimination metric, since that allows us to see how well our system can discern between positive and negative candidates. The secondary guiding metric was recall. In recruiting, it is better to accept a false positive (accept an unqualified applicant) than reject a false negative (reject a qualified candidate).

Why this is difficult. Because candidates are recruiter-sourced rather than drawn from an open applicant pool, most resume–job pairs already satisfy baseline relevance. The task therefore requires discriminating among many plausible candidates competing for limited interview capacity, not filtering obviously unqualified applicants.

4.2 Task 2: Resume-to-K Job Recommendation

Problem. Given a resume $r \in \mathcal{R}$ and a candidate-specific pool of jobs $\mathcal{J}(r) = \{j_1, \ldots, j_N\}$, the goal is to rank jobs by predicted fit. A system assigns a score s_{r,j_k} to each job and produces a ranking by sorting scores in descending order.

Relevance. A job j is considered relevant for resume r if $y_{r,j} = 1$, i.e., the candidate advanced to an interview stage for that job (Section 3.2).

Metrics. We evaluate ranking quality using Precision@K, Recall@K, and AUC-ROC.

4.3 Evaluation Protocol and System Comparison

Evaluation setting. We evaluate on (i) the RecruitBench production dataset (1,594 resume–job pairs across 18 roles) and (ii) a synthetic stress-test suite (26 cases) with predefined expected score intervals (Appendix/Table 7). We do not fine-tune model weights on RecruitBench; evaluation is performed by running each system in an inference setting with fixed prompts and fixed retrieval.

Synthetic stress-test metric. Each synthetic resume–job pair specifies an expected score interval $[L, U]$ derived from predefined scoring principles. A prediction s is considered consistent if $L \leq s \leq U$, and we report the fraction of synthetic cases that fall within their expected ranges.

Scoring architectures compared. For task 2, a key variable we study is how the scoring prompt is structured when evaluating a resume against many jobs. In the single-call flow, a single prompt scores and re-ranks 18 jobs in one LLM invocation, which can induce context bloating and "lost-in-the-middle" effects. In the parallel batched flow, the job pool is split into smaller batches (batch size six), each batch is scored in parallel, and results are aggregated into a final ranked list.

Latency measurement. In addition to accuracy and calibration metrics, we measure end-to-end latency from request initiation to producing the final ranked output. We report mean latency and tail percentiles (P50, P90, P95, P99) under identical model settings and concurrency constraints, since the primary motivation for batching is to reduce scoring latency without changing retrieval.

5 Results

We report results on the RecruitBench test split (470 resume–job pairs across 18 jobs. All methods—including classical screening baselines and LLM-based pipelines—are evaluated on the same test set. We report AUC-ROC computed globally over the full test set as the primary discrimination metric, along with Precision, Recall, and F1.

5.1 Task 1: Interview Advancement Prediction

5.1.1 Classical screening baselines

Table 4 summarizes non-LLM screening baselines on the test set. Most lexical overlap methods have limited discriminative power: TF–IDF achieves AUC-ROC of 0.483 and BM25 achieves 0.510, near chance-level ranking. Resume length performs worse than random (AUC-ROC 0.467), indicating that verbosity is not predictive of interview advancement. The strongest non-LLM baseline is an ATS-style keyword ruleset (AUC-ROC 0.545), but it remains far from reliable discrimination.

A notable pattern is that F1 can be misleading under this label distribution: because the positive rate is approximately 30% on the test set, high-recall strategies can achieve superficially competitive F1 even when discrimination is weak. We therefore emphasize AUC-ROC as the primary signal of ranking quality independent of threshold.

Method	Threshold	Precision	Recall	F1	AUC-ROC
Accept All	0.50	0.311	1.000	0.475	0.500
Random	0.50	0.315	0.508	0.389	0.510
Resume Length	6.20	0.282	0.446	0.345	0.467
TF–IDF Cosine	0.01	0.314	0.879	0.463	0.483
BM25	7.00	0.312	0.998	0.476	0.510
ATS Keyword Baseline	0.46	**0.318**	0.903	**0.470**	**0.545**

Table 4: Task 1 screening baselines on the RecruitBench test set.

LLM-based screening showed significant improvements in AUC-ROC. Prompt-only LLM scoring already substantially outperforms lexical baselines (AUC-ROC 0.620 vs. 0.545 for the best non-LLM baseline), indicating that LLMs extract higher-order signals from resume and job text beyond keyword overlap.

Adding job-specific context improves further. Retrieval of similar previously accepted candidates (RAG) provides a modest lift over prompt-only (0.630 vs. 0.620). A hiring-manager persona constructed from the transcript conversation yields a larger lift (0.660). Combining both signals—RAG plus persona—achieves the strongest overall discrimination (AUC-ROC 0.685), while accepting the smallest fraction of candidates (53%), indicating improved selectivity.

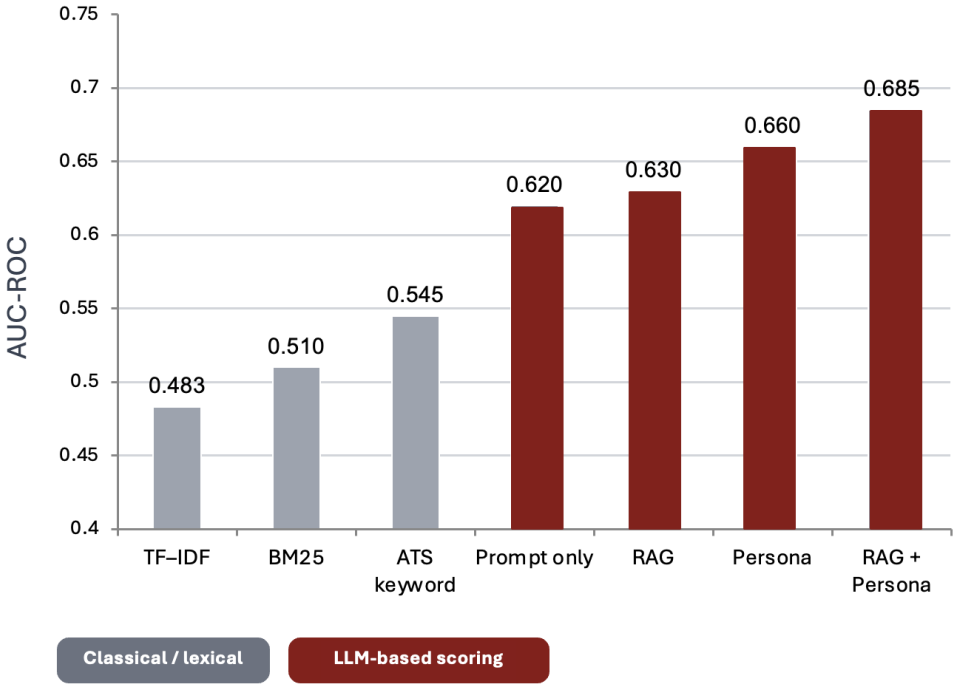

Figure 1: AUC-ROC across various experimental baselines.

5.1.2 Heterogeneity across jobs

Despite improvements from LLM-based methods, performance varies substantially across jobs. For example, prompt-only screening exhibits the largest per-job variance (per-job F1 standard deviation

0.228; range 0.000–0.842), including at least one job where it fails entirely (F1=0.000). Adding job-conditioned context reduces variance: the persona-based method has the lowest reported per-job F1 standard deviation (0.171), indicating more stable calibration across heterogeneous roles.

We also observe systematic differences by job type: LATAM roles consistently achieve higher F1 across all LLM methods (e.g., 0.583–0.647), while GTM roles exhibit larger gains from persona framing in AUC-ROC, suggesting that job-specific evaluative criteria are less recoverable from generic priors alone.

5.2 Synthetic Stress Tests

We evaluate robustness on a 26-case synthetic suite designed to probe controlled variants, edge cases, and adversarial resumes. All four LLM methods correctly reject all 7 adversarial cases (7/7), indicating robustness to straightforward resume gaming tactics such as keyword stuffing and credential inflation. Across the full synthetic set, RAG+Persona achieves the highest AUC-ROC (0.980) and perfect recall at its optimal threshold, while prompt-only and RAG screening achieve perfect precision at their optimal thresholds but miss one positive case (recall 0.889).

Method	Precision	Recall	F1 (optimal)	AUC-ROC	Adversarial Correct
Prompt Only	1.000	0.889	0.941	0.954	7/7
RAG Screening	1.000	0.889	0.941	0.938	7/7
Persona (no RAG)	0.818	1.000	0.900	0.974	7/7
RAG + Persona	0.818	1.000	0.900	**0.980**	7/7

Table 5: Synthetic stress-test performance (26 cases). "Optimal" indicates the best threshold for each method on this synthetic set.

5.3 Task 2: Resume-to-K Job Recommendation (Ranking)

Task 2 evaluates retrieval-conditioned job recommendation. We study how LLM scoring architecture affects ranking-relevant behavior under identical retrieval: (i) a *monolithic* single-call flow that scores the full pool at once, and (ii) a *parallel batched* flow that splits the pool into batches, scores batches concurrently, and aggregates scores into a final ranking.

5.3.1 Production screening adherence

As a production-aligned proxy for recommendation quality, we measure whether jobs known to be a good fit from production signals receive a score above the display threshold (≥ 7.0). On our benchmark, the parallel batched flow doubles recall at essentially unchanged precision (Precision: $0.284 \rightarrow 0.286$; Recall: $0.244 \rightarrow 0.512$) and improves AUC-ROC from 0.483 to 0.567. This indicates that the monolithic flow under-scores many true positives in large-context prompts, while batching improves discrimination at the cutoff.

5.3.2 Synthetic business-rule adherence

Outcome logs do not isolate failure modes like keyword stuffing or formatting sensitivity. We therefore evaluate a synthetic suite with predefined expected score intervals. On 26 synthetic cases, the parallel batched flow lands within the expected range more often than the monolithic flow ($34.1\% \rightarrow 41.3\%$), indicating improved calibration to explicit scoring rules.

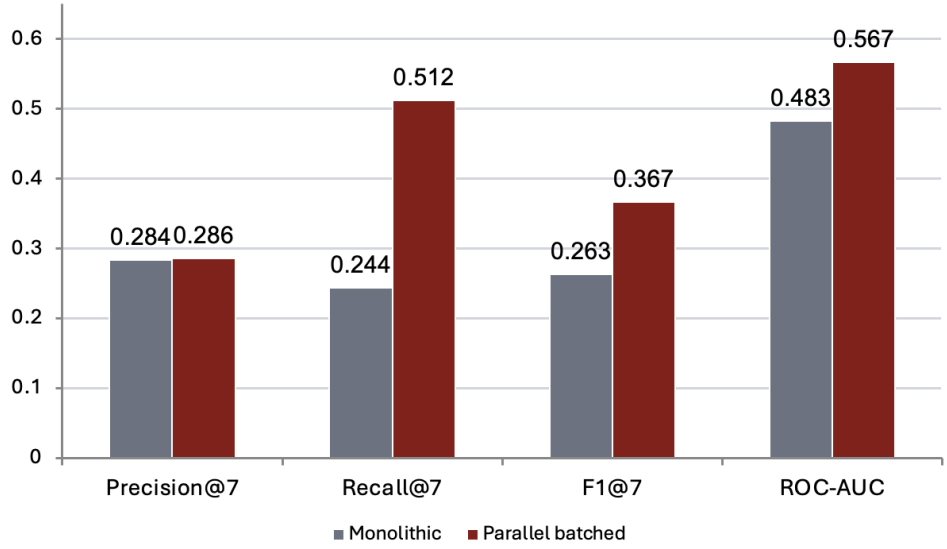

Figure 2: Parallel batching substantially increases recall and AUC-ROC at essentially unchanged precision.

5.3.3 Latency

Parallel batching improves end-to-end latency across the distribution. Over 600 runs, mean latency decreases from 48.3s to 25.6s and median latency (P50) decreases from 49.0s to 25.8s; tail latency also improves (P99: 64.6s $\rightarrow$ 30.5s). Notably, the parallel flow's P99 (30.5s) is faster than the monolithic flow's median (49.0s).

Latency Metric (600 runs)	Monolithic	Parallel Batched
Mean (ms)	48,276	25,619
P50 (ms)	49,043	25,780
P90 (ms)	57,533	28,723
P95 (ms)	60,073	29,623
P99 (ms)	64,553	30,516

Table 6: Task 2 end-to-end latency under fixed retrieval. Parallel batching yields $\sim$1.9$\times$ speedup across central and tail latency.

5.3.4 Qualitative preference

In a blind comparison judged by an independent frontier-LLM council over 600 entries, the parallel batched flow was preferred in 69.2% of cases versus 22.5% for the monolithic flow (8.3% ties), providing qualitative support that batched scoring produces better-ranked outputs under identical retrieval.

6 Discussion

Results show that RecruitBench can reliably differentiate recruiting systems across screening and recommendation tasks. We found substantial performance differences emerge between classical methods, LLM-based scoring approaches, and varying AI architectures.

For the screening task, traditional lexical and ATS-style baselines exhibited limited discrimination capabilities. These methods, like TF-IDF and BM25, performed near chance-level AUC-ROC on the interview advancement prediction task. This reflects the structure of the dataset: candidates are already sourced and screened by recruiters, so most resume–job pairs are plausibly relevant. The task requires distinguishing among candidates who appear qualified rather than filtering obviously irrelevant applications.

LLM-based scoring substantially improves discrimination. Even prompt-only scoring outperforms the strongest lexical baselines, indicating LLMs are able to reason beyond surface keyword overlap. Performance improves further when job-conditioned context is added. In particular, persona-based prompts yield the largest gains, suggesting that explicitly framing the evaluation from a hiring-manager perspective helps the model apply role-specific criteria. The combined RAG + persona configuration achieves the strongest screening performance while also reducing the acceptance rate, indicating better selectivity.

Results from the recommendation task highlight the importance of scoring architecture. When evaluating pools of candidate–job matches, a monolithic prompt that scores all jobs simultaneously tends to under-score relevant matches. In contrast, a parallel batched architecture, where jobs are scored in smaller groups and aggregated, substantially improves recall while maintaining precision. This suggests that large-context prompts may introduce evaluation artifacts such as context bloating or "lost-in-the-middle" effects that suppress strong matches. Batching mitigates this issue by allowing the model to evaluate each job with more focused context.

The batched architecture also improves system performance along real-world dimensions. End-to-end latency is reduced by nearly a factor of two, with improvements across both median and tail latency. In real recruiting systems where models operate interactively within recruiter workflows, latency is as important as ranking quality.

Overall, the benchmark enables systematic comparison of recruiting models across two operational tasks: predicting interview advancement for resume–job pairs and ranking job opportunities for a candidate.

7 Limitations and Future Work

RecruitBench represents an initial benchmark for evaluating AI systems in recruiting workflows, but several limitations remain.

The current benchmark is relatively small and concentrated in a limited set of startup roles. While this allows controlled evaluation, the results may not generalize to other labor markets, industries, or hiring pipelines. Expanding the dataset to include additional roles, companies, and regions would improve the robustness of future evaluations.

Second, hiring outcomes aren't purely objective assessments of candidate quality. Factors such as recruiter incentives, internal company politics, and external market conditions can influence which candidates advance through pipelines.

Future work could incorporate continuous feedback into our AI systems. Recruiting decisions unfold across multiple timesteps where candidates are rejected and accepted over a period of time. It would be interesting to see how we can evaluate systems that learn from past candidates and

evolve to be job-specific.

Finally, fairness analysis remains an important open question. As automated screening tools become more prevalent, it will be important to evaluate whether models behave consistently across different candidate populations and hiring contexts. While RecruitBench is grounded in real world decisions, that doesn't imply a lack of bias.

References

[Barocas and Selbst(2016)] Solon Barocas and Andrew D. Selbst. Big data's disparate impact. *California Law Review*, 104(3):671–732, 2016.

[Bian et al.(2020)Bian, Li, et al.] Yun Bian, Yukun Li, et al. Multi-view co-teaching for job-resume matching. *arXiv preprint arXiv:2009.13299*, 2020.

[de Ruijt et al.(2021)] Paul de Ruijt et al. Job recommender systems: A review. *arXiv preprint arXiv:2111.13576*, 2021.

[Geyik et al.(2018)Geyik, Kenthapadi, et al.] Sahin Cem Geyik, Krishnaram Kenthapadi, et al. Talent search and recommendation systems at linkedin. In *Proceedings of KDD*, 2018.

[Kenthapadi et al.(2017)] Krishnaram Kenthapadi et al. Personalized job recommendation system at linkedin. In *Proceedings of RecSys*, 2017.

[Liu and de Rijke(2024)] Yuqing Liu and Maarten de Rijke. A survey on robustness in information retrieval. *ACM SIGIR Forum*, 2024.

[Maheshwary et al.(2018)Maheshwary, Anand, et al.] Rishabh Maheshwary, Saket Anand, et al. Matching resumes to jobs via deep siamese network. In *Proceedings of WWW Companion*, 2018.

[Manning et al.(2008)Manning, Raghavan, and Schütze] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[New York City Department of Consumer and Worker Protection(2023)] New York City Department of Consumer and Worker Protection. Automated employment decision tools (local law 144) guidance. NYC Government Documentation, 2023.

[Organizers(2016)] RecSys Challenge Organizers. Recsys challenge 2016: Job recommendations on xing. In *Proceedings of RecSys*, 2016.

[Raghavan et al.(2020)Raghavan, Barocas, Kleinberg, and Levy] Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. Mitigating bias in algorithmic hiring. In *Proceedings of FAT* (now FAccT)*, 2020.

[Ribeiro et al.(2020)Ribeiro, Wu, Guestrin, and Singh] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of ACL*, 2020.

[Robertson and Zaragoza(2009)] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4): 333–389, 2009.

[U.S. Equal Employment Opportunity Commission(2023)] U.S. Equal Employment Opportunity Commission. Assessing adverse impact in software, algorithms, and artificial intelligence used in employment selection. EEOC Technical Assistance Document, 2023.

[Yu et al.(2024)] Yue Yu et al. Confit: Contrastive learning for resume–job matching with data augmentation. In *Findings of ACL*, 2024.

A Synthetic Dataset

Category	Case	Description	Range
Controlled Variants (8)			
Controlled	Perfect Match	Exceeds all requirements; ideal companies; strong trajectory	[8.5, 10.0]
Controlled	Strong Match	Meets requirements; lacks ideal pedigree	[7.0, 8.5]
Controlled	Skills Only	Skills + YOE but no quality signals	[4.5, 6.0]
Controlled	Wrong Domain	High pedigree, unrelated functional domain	[3.5, 6.0]
Controlled	Hard Req Miss	Missing primary hard requirement	[2.5, 4.5]
Controlled	Overseas (US Role)	Unknown overseas background for US role	[3.0, 5.0]
Controlled	Overqualified Manager	Director/VP applying for IC role	[3.0, 5.5]
Controlled	Junior for Senior	¡2 YOE applying for 5+ YOE role	[1.0, 3.0]
Edge Cases (11)			
Edge	Career Changer	Finance/quant transitioning to SWE	[3.5, 6.0]
Edge	YC Founder	Non-traditional founder with strong signals	[7.0, 9.0]
Edge	Padded Resume	Buzzword-heavy, low substance	[3.0, 5.5]
Edge	LATAM (Regional Fit)	Strong LATAM engineer for LATAM role	[7.0, 9.0]
Edge	LATAM (US Role)	Strong LATAM candidate, no US authorization	[3.0, 5.5]
Edge	Career Gap	Ex-FAANG with 2-year gap	[6.0, 8.5]
Edge	Ideal Company Match	Exact match to job's ideal companies	[8.5, 10.0]
Edge	All Green Flags	Matches all positive hiring signals	[6.0, 8.0]
Edge	All Red Flags	Matches all red flags	[3.0, 5.5]
Edge	Format Plain	Clean single-column resume	[7.0, 9.0]
Edge	Format Fancy	Identical content, complex layout	[7.0, 9.0]
Adversarial Cases (7)			
Adversarial	Keyword Stuffing	Large skill list without evidence	[1.0, 3.0]
Adversarial	Date Inflation	Overlapping roles exaggerating YOE	[2.0, 5.0]
Adversarial	Implausible Metrics	Unrealistic leadership/revenue claims	[2.0, 4.5]
Adversarial	Irrelevant Prestige	Impressive but wrong domain	[1.0, 3.0]
Adversarial	AI-Generic Resume	Buzzword-heavy, unverifiable claims	[2.0, 4.5]
Adversarial	Corporate IT Mismatch	Consulting IT for startup role	[2.0, 4.5]
Adversarial	Job Hopper	7 roles in 6 years	[3.0, 5.5]

Table 7: Synthetic scenario taxonomy in RecruitBench. Each case isolates a specific scoring dimension to evaluate calibration and robustness.