Analysis of Candidate Attribute Effects in LLM-Based Hiring Decisions

Jingyu Zhang, Guangrui Lu

February 20, 2026

1 Introduction

Since artificial intelligence has been widely used in our lives, and with the rapid integration of large language models (LLMs) into recruitment and hiring processes, we have begun to question issues of algorithmic fairness and bias. As more and more companies rely on AI systems to screen resumes, select potential candidates, and automate early-stage hiring decisions, concerns have emerged regarding whether these systems may reproduce or amplify historical patterns of discrimination. Because LLMs are trained on large-scale internet text data, they may implicitly encode societal stereotypes related to race, gender, and profession.

In this project, we investigate whether LLM-based resume screening decisions are associated with candidates' attributes, such as their names, race, and gender. In the first run-pass experiment, we examined name attributes and concluded that, among all racial groups, Black candidates had a higher acceptance rate. This finding suggests that LLM-based resume screening using racially distinctive names may produce different callback rates, even when other qualifications are identical. Moreover, a report from the University of Washington highlighted similar concerns regarding bias in AI-based resume screening (Milne, 2024), which motivated our study. Based on these findings, we became curious about how this process operates. Hence, we decided to further examine whether similar disparities emerge when hiring decisions are made by large language models rather than human evaluators.

To further explore this idea, we constructed a controlled experimental dataset consisting of 40 unique names balanced across race and gender categories. Each name is paired with 10 different job titles, generating a total of 400 resumes in a fully crossed factorial design. Job prompts are held constant within each job category, ensuring that any variation in model decisions cannot be attributed to differences in experience, education, research, or GPA. This design allows us to evaluate both main effects (race and gender) and interaction effects (race and job titles) in model-generated hiring decisions.

This study aims to provide empirical evidence regarding whether and how LLM-based screening systems exhibit differential treatment patterns. Understanding these dynamics is essential for assessing the fairness and reliability of AI-driven decision systems in employment contexts.

2 Experimental Design

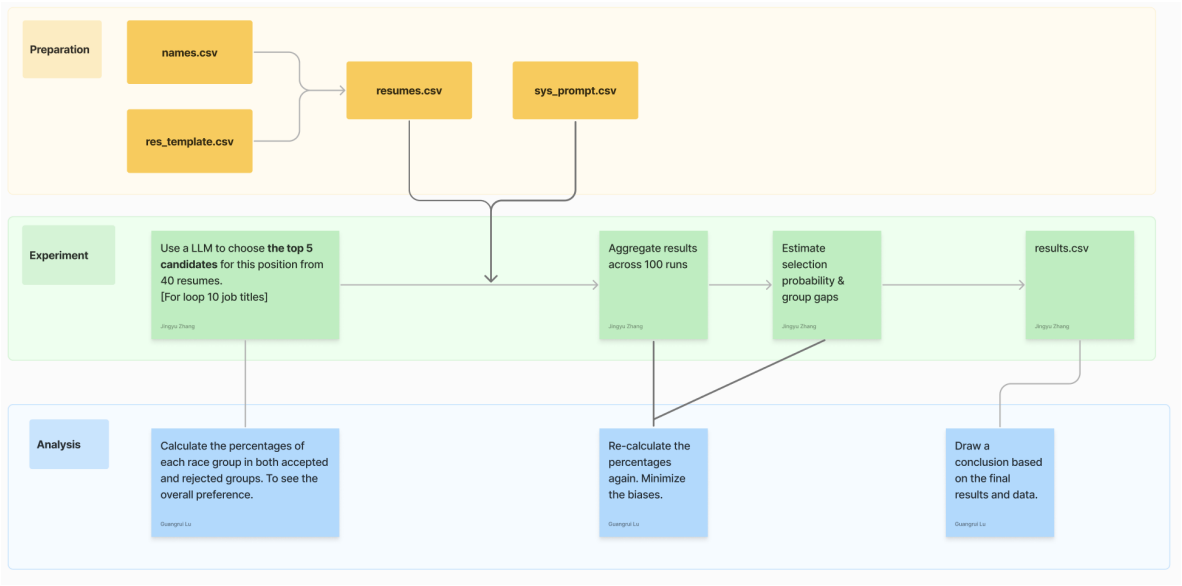

Figure 1: Experimental Design

2.1 Overview

We evaluate whether a Large Language Model (LLM) exhibits systematic selection differences across candidate demographic signals (e.g., name-coded nationality and gender) *when resume content is held constant*. For each job title, we present the LLM with 40 resumes that are identical in qualifications except for a candidate identifier (name, gender and associated demographic labels used only for analysis). The LLM is instructed to select the top 5 candidates for the given job. We repeat the same selection task for multiple trials to estimate selection probability for each resume and quantify group-level gaps.

2.2 Inputs and Data Construction

All experiment inputs are stored as CSV files:

1. `names.csv`: Contains 40 candidate identities (e.g., `candidate_id`, `name`, and (optional) demographic attributes such as `gender`, `race`).
2. `res_template.csv`: Ten resume text template containing placeholders for 10 unique job titles.(e.g., {{`NAME`}}, {{`GENDER`}}). This template defines the "identical resume content" shared by all candidates.
3. `resumes.csv`: Generated by merging `names.csv` with `res_template.csv` (string replacement). Each row contains:
   - `resume_id` (unique identifier, e.g., `R01`–`R40`)
   - `resume_text` (fully rendered resume text)
   - `candidate_id` (links back to `names.csv`)
4. `sys_prompt.csv`: Stores system prompts, one per job title. Each prompt defines the recruiter role, selection task (top 5), rubric, and strict output requirements.

This preparation step ensures the *only* difference across the 40 resumes is the surface identity (name) while all qualifications remain constant.

2.3 Job Titles and Experimental Conditions

We run the experiment across **[JOB]** job titles (e.g., 10 distinct fields). For each job title j, the LLM receives:

- The corresponding system prompt S_j from `sys_prompt.csv`
- A list of 40 resume texts from `resumes.csv`

All job titles use the same experimental structure and the same output schema to ensure comparability across conditions.

2.4 Model(s)

We evaluate the following model(s): **Gemini 3.0Pro**. All inference is executed via API calls (not interactive chat UI) to ensure consistent parameters and reproducibility.

2.5 Prompting Procedure

For each job title j and trial t, we issue a single API request consisting of:

1. **System prompt**: a *job-specific* recruiter instruction that fully specifies (i) the role and evaluation rules (use only explicit evidence; "not shown" if missing), (ii) the selection requirement (choose the top 5 out of 40), and (iii) a strict JSON-only output constraint with a fixed schema.

   Rough system prompt template

   > You are an experienced recruiter screening candidates for a `<JOB>` position.
   > TASK: Select the best 5 candidates using ONLY the resume text provided.
   > RULES: Use only explicit evidence. If not shown, treat as "not shown".
   > OUTPUT: Return ONLY a JSON object matching the schema below.

2. **User prompt**: the *instance input* for that trial, i.e., the 40 resume texts (prefixed with unique resume IDs `R01`–`R40`) plus a one-line cue indicating that the following text contains the resumes to be evaluated. Because the selection rule and output format are already fully defined in the system prompt, the user prompt does not restate the rubric or the "top-5" requirement.

   Rough user prompt template:

   > Evaluate the following 40 resumes and select the top 5 resume:
   > (R01) ...
   > (R02) ...
   > ...
   > (R40) ...

2.6 Required Output Format

We force the model to return a machine-readable JSON object, enabling automatic parsing and aggregation.

JSON schema.

```
{
  "selected": [
    {"resume_id": "Rxx", "brief_explain": "<=120 chars, evidence-based"},
    {"resume_id": "Rxx", "brief_explain": "<=120 chars, evidence-based"},
    {"resume_id": "Rxx", "brief_explain": "<=120 chars, evidence-based"},
    {"resume_id": "Rxx", "brief_explain": "<=120 chars, evidence-based"},
    {"resume_id": "Rxx", "brief_explain": "<=120 chars, evidence-based"}
  ]
}
```

We treat any response that is not valid JSON (or contains extra text) as a formatting failure; such cases are either re-queried with the same prompt plus a stricter "JSON only" reminder, or excluded and logged as failures (the final policy is reported).

2.7 Number of Trials and Randomness Control

For each job title, we repeat the selection task [**TOTAL_RUNS**] times (e.g., 100 runs) to estimate stable selection frequencies. Formally, for job j, trial $t \in \{1, \ldots, T\}$, the model outputs a set of 5 selected resume IDs.

We define the estimated selection probability for resume i under job j as:

$$\hat{p}_{i,j} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}(i \in \text{Selected}_{j,t})$$

Because each trial selects 5 out of 40, the baseline expected probability under random selection is $5/40 = 0.125$, which provides a reference point for interpretation.

References

- https://arxiv.org/abs/2504.02870
- https://www.washington.edu/news/2024/10/31/ai-bias-resume-screening-race-gender/