Project Design:
The High-Quality Job Hybrid Recommendation System
Based on LinkedIn Datasets

Name: Hang Pan zID: z5598515

1. Scope

1.1 Domain and Users:

Nowadays, a huge number of graduates find it hard to get suitable jobs and remain unemployed all the time, and employers also have difficulty finding satisfactory employees. The Hybrid Recommend System focus on the job recruitment industry, specifically designing a hybrid job recommendation system that provides personalized job recommendations to users. Target users include: Job seekers, like University students and graduates who searching for internships or new positions; Mid-career candidates exploring career transitions; Passive candidates who are willing to be handpicked for opportunities. For employers and recruiters, who aim to attract qualified candidates. While the current system prototype emphasizes recommendations to users, the system architecture allows extension to recommend candidates to employers as well.

The system is built using the LinkedIn Job Postings datasets (Kaggle, 2023) and Resume datasets (Kaggle, 2022) to ensure the recommendations are realistic and broadly representative outputs.

1.2 Presentation Format & Interface:

During the system initialization phase, users are required to enter certain fields, such as positions of interest, salary range, etc. The main function is to solve the cold start problem. At main function page, the system will display 4 – 8 job postings on a responsive web interface, every job card displaying the title of job, company name,

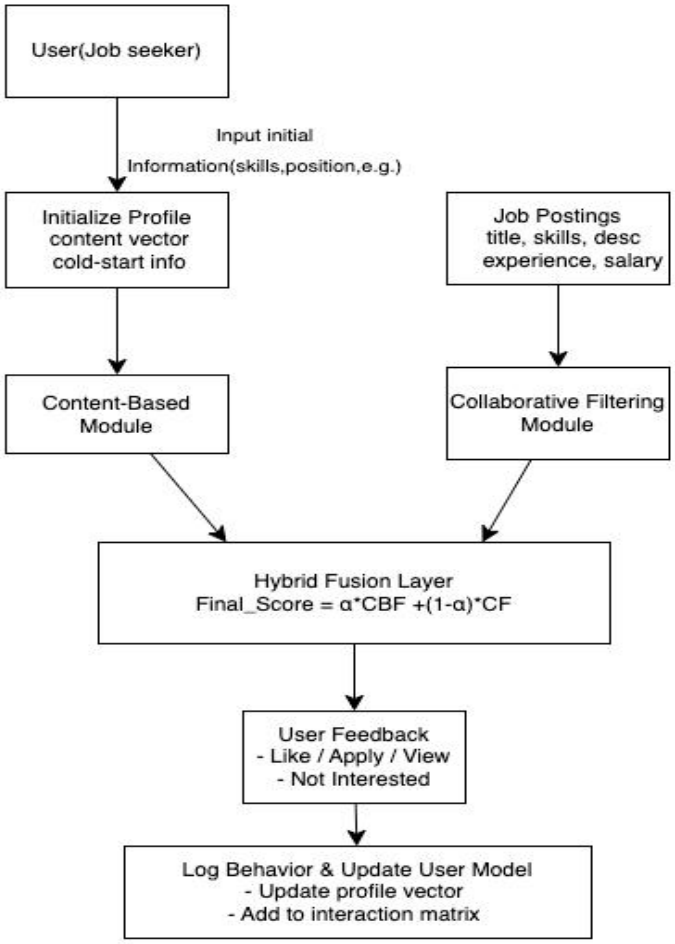
location, job description, salary range, skills required, save button and apply button. Users interact with the interface by: clicking "Interested" (positive feedback), clicking "Not Interested" (negative feedback), seeing full job details, saving or applying for a job. Users interact by clicking buttons and reading job descriptions.

1.3 Simple diagram of user interface and description of interaction:

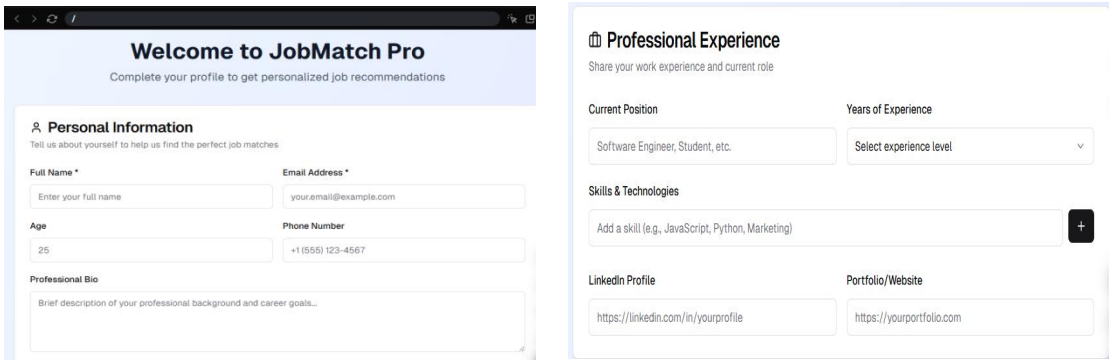
The system will simulate implicit and explicit user feedback:

1. Explicit feedback: Interested/Not Interested, Save, Apply;
2. Implicit feedback: Job viewed, time spent on page.

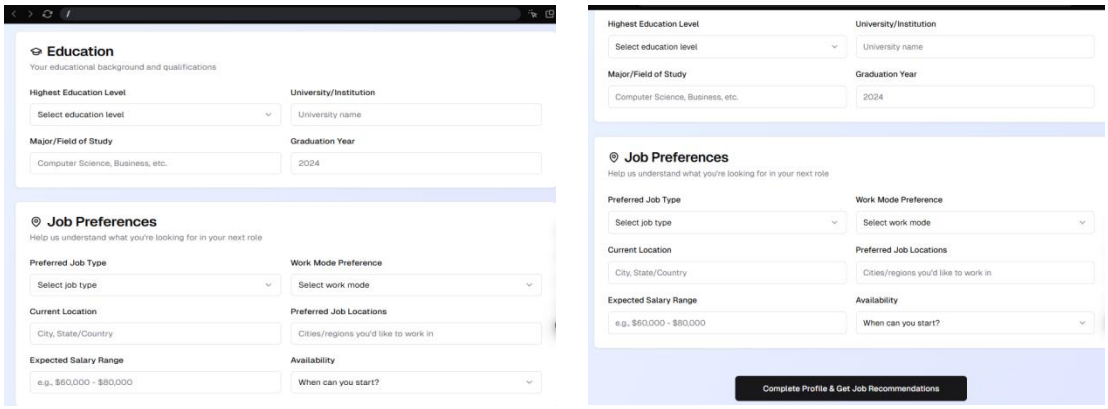
These actions will be logged and incorporated into the recommendation engine, which updates dynamically to reflect user preferences. Here, the system flow and some system demo pages are shown:



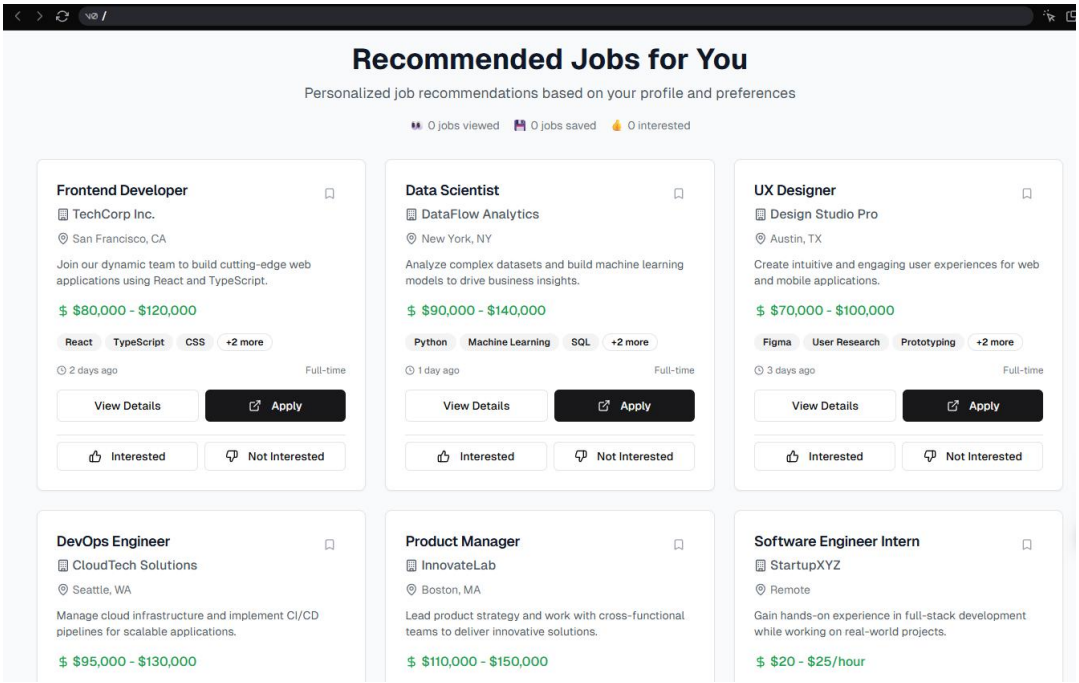
1.1 Diagram of system



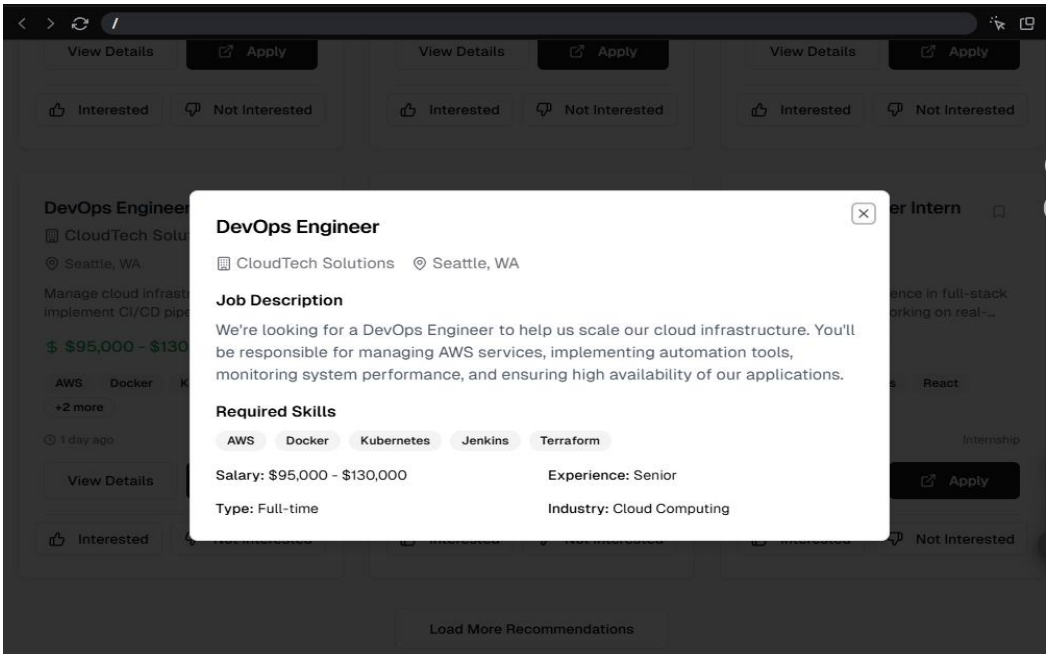
1.2 sign up for personal information(1)



1.2 sign up for personal information(2)



1.3 home page for recommendation



1.4 some of job details

1.4 Dynamic System & Cold Start via Hybrid Approach:

To handle dynamic updates, the recommendation model will be periodically updated using the latest user interaction data. For example, the recommendation engine is updated every three or seven days.

The cold start problem will be addressed using a hybrid recommendation strategy: For new users: content-based filtering based on user-inputted profile (skills, education, experience). For new jobs: use job content (skills, industry, title) to find similar jobs via embedding or TF-IDF algorithm.

As user behavior accumulates, collaborative filtering gradually enhances accuracy. These approach ensures early-stage personalization without waiting for extensive user history.

1.5 Business Model and Revenue Generation

In a business scenario, the system can generate revenue in the following ways: (1)Sponsored jobs: Companies pay to increase job visibility and attract job seekers' attention. For example, if 500 companies pay $100/month for sponsored listings, this alone yields $50,000 per month in recurring revenue. (2)Premium user plan: Provides enhanced features, such as the ability to view more suitable job

recommendations, AI-driven resume matching, priority application queues, directly talk with recruiter,and user profiles built every seven days for ordinary users and every three day for premium users. If just 5% of 100,000 users subscribe at $9.99/month, monthly revenue exceeds $49,000. (3)Affiliate partnerships: Earn profits by extracting commissions from successful job application conversions.

This is a platform that can become an HR tech product worth multi-millions within 2–3 years since launch, especially in highly job seeker-concentrated and digitally hiring-adopting markets.

2. Datasets

This project uses two realistic and complementary datasets from Kaggle to support the development of a hybrid job recommendation system. These datasets simulate a LinkedIn-style platform with both job listings and user resume information, enabling the modeling of interactions between job seekers and job postings. After data selection, field selection and data pre-processing, the following datasets are obtained:

2.1 Job Datasets

Files used:
job_postings.csv: Contains structured information about job opportunities, including:
job_id, company_id, company_name, title(Job title), description(Job description), max_salary,location, formatted_work_type(Fulltime, Parttime, Contract), skills_desc, application_url, views(Number of times the job posting has been viewed), original_listed_time(Original time the job was listed)

companies.csv: Maps each job to a company, with metadata:
company_id,name, company_size(Company grouping based on number of employees), country, state, address, description(Company description)

employee_counts.csv: Gives the number of employees per company:
company_id, employee_count, follower_count

This dataset provides the content features required for job representation, such as skills_desc, description, job types (Fulltime, Parttime, Contract), and location. These fields are essential for content-based filtering, as well as for modeling job popularity (using views) and job availability over time (via original_listed_time). For metadata fields (like experience level, work type, remote_allowed) used for filtering or context-aware ranking.

2.2 Resume Datasets

File used:
resume_data.csv:Simulates job seeker profiles with rich textual and structured data: user_id, address, career_objective,skills, educational_institution_name,degree_names, responsibility, passing_years_educational_results, major_field_of_studies, positions, locations, resume_text,job_position_name, matched_score (The score of the matching degree between the user and a certain target position)

This datasets enables the creation of detailed user profiles. For example:
1.Extracting key skills and degrees for profile-job matching.
2.Using resume_text and career_objective for embedding-based semantic comparison.
3.Simulating user interest or preferences based on positions, locations.
4.Matched_score for historical user-job interaction, like apply or not apply, which to support collaborative filtering.

2.3 Datasets Limitations:

Despite being rich and well-structured, both datasets have limitations:
The job datasets may not include user interaction history (like clicks, applications), requiring simulation or hybrid approximation.
The resume datasets does not contain real-time behavior data (such as actual job applications), so preferences must be inferred from static profile content.
As both datasets are curated and possibly sanitized, they may not capture the full complexity, noise, or diversity of real-world recruitment platforms.

2.4 How to use these data:

By integrating these two datasets, the system can simulate realistic job recommendations by:

Matching job requirements (skills, degree, location) with user profiles.

Using user profile embedding from resume_text, career_objective to match with description in job postings.(via TF-IDF / BERT).

Supporting hybrid recommendation by combining profile-job similarity (content-based) with simulated interest labels.

Simulate implicit preferences via job_position_name and matched_score.

3. Method

To effectively recommend jobs to users based on their resumes and inferred preferences, I propose a hybrid recommendation system that combines both content-based and collaborative filtering approaches. This hybrid design enables the system to perform well in both cold-start and data-rich scenarios.

3.1 Content-Based Filtering (CBF)

We first implement a content-based recommendation method that matches user profiles with job descriptions.

User representation: Extracted from resume_data.csv, including skills, degree, past roles, and semantic embeddings of the resume text and career objective (using TF-IDF or BERT).

Job representation: Constructed from job_postings.csv, including required skills, job description, skills_desc, location, formatted_work_type and work type.

Matching mechanism: Use cosine similarity between user and job vectors to rank relevant jobs.

This is because, the CBF works well for cold-start users without prior interaction history and leverages the rich resume and job text data available.

3.2 Collaborative Filtering (CF)

To simulate user behavior data, we may either: using implicit application/view data (if present) or simulate historical interactions using resume-job compatibility labels (“match” vs. “non-match” based on category/skills).

We will experiment with:

Item-based collaborative filtering (via similarity in job views or applications)

Matrix factorization (using SVD) to learn latent user – job preference patterns

This is because the captures latent patterns from user interaction behavior and helps in personalizing recommendations based on user community similarity.

3.3 Hybrid Recommendation

To get the best recommendation, we propose a hybrid method that combines content-based score and collaborative filtering scores:

Final Score = $\alpha \times$ cb_score + $(1 - \alpha) \times$ cf_score. For weight α in [0,1], can be tuned on validation set using grid search.

This fusion have several advantages:

This can have good address in cold-start conditions and adapt to user feedback.

Context-aware filtering: Using work type (Full-time, Part-time,), location, or time of posting as context dimensions.

Semantic similarity is computed by applying BERT-based embeddings on job descriptions and resume texts, following the successful use of BERT for semantic representation in employee-related contexts, which enables better representation and understanding of the text and labels (Hemnath, 2022).

These three methods are implementable with the available datasets because the resume and job datasets provide rich information that can be vectorized and matched. While interaction data is not directly available, user–job compatibility can be simulated using profile-job matching, enabling the use of collaborative filtering. Combining both in a hybrid strategy allows the system to handle both new users and experienced users, supporting personalization and cold-start robustness.

4. Evaluation

To assess the performance and quality of the hybrid job recommendation system, we will conduct both offline evaluations using historical or simulated data and user-focused evaluations through interface simulation and feedback collection. This

two-pronged strategy ensures a well-rounded understanding of both algorithmic effectiveness and user satisfaction.

4.1 Offline Evaluation (Model Performance)

We will evaluate the ranking quality of our recommendation system using standard Top-N recommendation metrics:

Precision@K: Measures the proportion of recommended jobs in the top K that the user is likely to apply for.

Recall@K: Measures how many of the jobs the user was interested in are actually recommended.

F1@K: Harmonic mean of Precision and Recall.

MAP (Mean Average Precision): Considers ranking and relevance across users.

NDCG@K (if time permits): Takes ranking position into account.

Data split strategy:

The interaction dataset (applications or simulated matches) will be split using 80% train, 10% validation and 10% test strategy.

In absence of real interaction data, we simulate ground truth by matching resumes and job categories or tags.

4.2 User Simulation and Study (System-Level Evaluation)

A lightweight simulated user interface will be used to gather user feedback. This mock interface displays a top-N list of jobs with basic user interactions: “Interested”, “Not Interested”, “view details”, “save”, and “apply”.

We will recruit test users (e.g., classmates) to interact with the system and fill out a post-task questionnaire.

Feedback to be collected includes:

1. Perceived relevance of job recommendations
2. Diversity and novelty
3. User satisfaction
4. Intention to apply or save

Collected feedback will be analyzed using:

1. Likert-scale survey questions (1 – 5) for subjective evaluation
2. Click-through rate (CTR) or interaction count for behavioral analysis

4.3 Trade-offs and Model Selection

In case of multiple candidate models (baseline CBF, CF, or weighted hybrid), we will compare them using:

1. MAP or F1@K as primary metrics
2. Diversity and novelty as secondary metrics
3. User preference feedback as qualitative support.

Model selection will be based on overall balance between accuracy and user satisfaction.

4.4 Computation and Practical Constraints

All models are designed to work in batch inference mode, meaning recommendations are precomputed and displayed in real time. In the future, we will explore incremental updates using new user interactions. To further improve scalability and efficiency, we can use a vector cache system, such as FAISS to store dense vector embeddings. This reduces response time from seconds to milliseconds for content-based filtering. To

further improve cold-start experience, using rule-based bootstrapping to recommend jobs base users' location.

References

Hemnath, R. Integrating Natural Language Processing with BERT and LSTM for Employee Sentiment Analysis in HRM.

Kaggle. (2023). LinkedIn Job Postings Dataset (2023–2024) [Data set]. Kaggle. https://www.kaggle.com/datasets/arshkon/linkedin-job-postings/data

Kaggle. (2022). Resume Dataset [Data set]. Kaggle. https://www.kaggle.com/datasets/iamsouravbanerjee/resume-dataset