Personalized Job Recommendation

APAN 5430 - Group 4

Minkyung Kim | Boni Rosen | Kibaek Kim | Kas Kiatsukasem | Philip Suchakrey

Table of Contents

Business Background & Problem

In an **online job application portal** today, most recruitments are **relying on keyword-based search** and macthing where recruiters enter skill keywords or technical skill, and candidates are ranked only by those texts overlap.

This approch overlooks:

1. Candidate contextual relevance
2. Full story of candidate's background

System overlooks lead to:

1. Missed the right candidate opportunity
2. Poor overall match quality

Project Goal and Use Case

The goal of this project is to **develop a personalized job recommendation system** that:

- Calculates a **match score** between a candidate's resume and job descriptions
- **Rank job postings by compatibility**, showing the most relevant and well-aligned positions first, based on **both explicit qualifications and implicit signals**

Use case:

- **Candidate side**: Upload their resumes to receive a ranked list of job openings that best fit
- **Recruiter side:** Identify the most relevant candidates based on overall profile similarity.

Data source

The goal is to combine data from two sources:

- **LinkedIn Job Board:** Most up-to-date and realistic job postings
- **Hugging Face job dataset:** to ensure sufficient data volume and diversity of jobs listed

Web Scraping from Linkedin Job Board

- Scraped ~30K LinkedIn job posts using linkedin_jobs_scraper
- 100 predefined titles (e.g., Digital Marketing Specialist)
- Filtered for U.S., remote, mid–senior, $100K+, last 30 days
- Headless Chrome with random 1–4 min pauses to avoid detection
- Extracted fields: Job Title, Company, Date, Link, Insights, Description
- Saved up to 10K records per JSON file (30K Total)

Job Description Data Set

The raw data source contained 22 columns with details about job opportunities. We adjusted the dataframe and filtered unnecessary columns. Finally, we have ~1GB data set filtered and includes only 10 important columns for the project

Source: https://huggingface.co/datasets/lof223/job_description

Linkedin

1. **Job Title - The title of the job position**
2. **Company - The company name**
3. **Company Link - URL link to the company**
4. **Date - The date the job was posted**
5. **Date Text - Text representation of the date**
6. **Job Link - URL link to the job posting**
7. **Insights - Additional insights about the job**
8. **Description Length - The character count of the job description**
9. **Description - The full text of the job description**

Hugging Face

```
df_all_jobs.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1615940 entries, 0 to 1615939
Data columns (total 11 columns):
 #   Column            Non-Null Count    Dtype 
---  ------            --------------    ----- 
 0   Job Id            1615940 non-null  int64 
 1   Experience        1615940 non-null  object
 2   Salary Range      1615940 non-null  object
 3   location          1615940 non-null  object
 4   Country           1615940 non-null  object
 5   Job Posting Date  1615940 non-null  object
 6   Job Title         1615940 non-null  object
 7   Company           1615940 non-null  object
 8   Job Description   1615940 non-null  object
 9   Benefits          1615940 non-null  object
 10  skills            1615940 non-null  object
dtypes: int64(1), object(10)
memory usage: 135.6+ MB
```

*Final columns in each data sources

Selected technologies & Rationale

01 Named Entity Recognition

Purpose:

To extract explicit, **factual entities** from resumes and job descriptions.

- Detects skills, company names, job titles, certifications, degrees, and other key entities.
- **Captures what the candidate knows or has done.**

02 Word/Sentence Embedding

Purpose:

To measure **contextual similarity** between Resume and JD text.

- Computes cosine similarity between the two embeddings to evaluate semantic closeness.
- **Captures what the candidate meant in the resume, even when the wording is different.**

03 Topic Modeling

Purpose:

To identify which **domain or industry topics** each document belongs to (e.g., Data Science, Finance).

- Trains an LDA model on JD text to learn latent topic distributions.
- Infers topic probabilities for each resume and compares them with each JD.
- **Captures whether both texts belong to the same industry or domain.**

Integrated Scoring Engine - Job Recommendation

- Each score measures a different dimension of “fit.”
- They are combined through weighted aggregation to compute the overall match score.

Final Job Matching Score = 0.45×SkillScore+0.35×SemanticScore+0.20×TopicScore

NER: Skill Score | **Word Embedding: Semantic Score** | **Topic Modeling: Topic Score**

Design Structure

1

Data Source

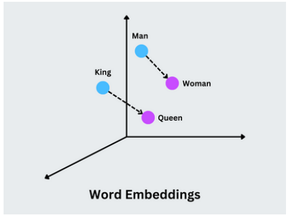
2

Preprocessing

3

NER
(Explicit Matching)

Word Embedding
(Contextual Matching)



Topic Modeling
(Domain Matching)

4

Final Job
Matching Score

5

UI

Streamlit

Evaluation Criteria & Methods

	Category	Metric	Description
Quantitative	System Performance	Query response time	Average query response time (e.g. ≤ 2 seconds per query) after input being fed into system (user's resume) and keyword search.
	Result Relevance	Query Precision	% of results matching user job search based on the final job matching score to check the relevancy and Mean Reciprocal Rank (MRR) to see whether the correct job appears early. Also to assess if the weighting in the final score has been set properly
Qualitative	User Experience (UX)	Ease of Interaction	Users can easily search and filter by job title, job description, and years of experiences.
	User Impact	Insight Feeding	The extracted information provides insights that could assist job seeking process and understand key requirements for specific role.

Thank you

APAN 5430 - Group 4