Semantic Graph-Based Job Recommendation System

Dhruv Anand
School of Computing and Augmented Intelligence
Arizona State University
danand6@asu.edu

Krishnaprasad Palamattam Aji
School of Computing and Augmented Intelligence
Arizona State University
kpalamat@asu.edu

Marie Rudasics
School of Computing and Augmented Intelligence
Arizona State University
mrudasic@asu.edu

ABSTRACT

In the current labor market, job searches are proving to be especially tough for many candidates searching for sufficient roles. Many job search algorithms do not always seem to match individuals with their best options. Utilizing techniques such as keyword matching or k-nearest neighbors are far too simple to capture the complexities that factor into an individual's career preferences. Our research introduces an innovative semantic job recommendation framework that combines three powerful technologies: BERT's contextual language understanding for rich text representation, FAISS (Facebook AI Similarity Search) for rapid vector similarity matching, and network-based recommendation refinement through Personalized PageRank. By integrating these complementary approaches, the system creates a comprehensive understanding of both job requirements and candidate preferences beyond what traditional systems can achieve. Our methodology transforms user queries into active nodes within a semantic job similarity network, enabling recommendations to be dynamically reranked based on the local graph structure rather than simple vector matching. Comparative evaluation against traditional KMeans clustering demonstrates our approach delivers superior performance across key metrics, providing recommendations that are both more contextually relevant and offer clearer reasoning behind each suggestion.

KEYWORDS

Semantic Matching, BERT Embeddings, Sentence-BERT (SBERT), FAISS, ANN Search, Personalized PageRank

1 Introduction

1.1 Background

Online job recruitment platforms such as Indeed, LinkedIn, and Glassdoor have utilized data science concepts to provide users with jobs that align relative to their preferences. Portals such as these provide hundreds of application links to individuals, providing a large influx of applications to companies. While it connects individuals with a vast number of job options, it also can be quite overwhelming as the quantity of jobs may be too high, and it also may not provide the best matches for an individual. Recruiters often find themselves dealing with a high quantity of applicants but a low quality of applications and match to the job description. The need to optimize and improve job search algorithms is greater than ever before, as numerous individuals are currently finding themselves with great difficulty in the job search process.

Job recommendation systems typically utilize keyword matching, user-behavior-based collaborative filtering, or rule-based filters. Limitations may occur, such as vocabulary mismatches, improper alignment on background qualifications, and inability to handle new users. In addition, job titles and descriptions can differ greatly across industries and companies, which provides even greater difficulties in the matching process.

Modern NLP architectures like BERT enable the encoding of complete job listings and candidate preferences into a unified vector space that captures deeper semantic relationships beyond simple keyword matching. These semantic embeddings facilitate efficient comparison through technologies such as FAISS, enabling scalable similarity-based search operations across large datasets.

The growing digitization of the labor market has led to an explosion in the availability of online job postings. These postings serve as rich sources of information for analyzing employment trends, salary distributions, demand for skills, and regional job availability. The dataset analyzed in this report consists of 10,000 job postings collected from various companies across the United States. Each entry provides detailed information including job titles, company names, descriptions, salary ranges, locations, work types, experience requirements, and other metadata.

Understanding this data is critical for identifying hiring trends, evaluating compensation structures, and supporting workforce planning. However, the dataset also contains inconsistencies and missing values, particularly in salary, skills, and application metrics, which must be addressed through data cleaning and preprocessing.

This report explores the structure and quality of the dataset and lays the foundation for further analysis such as job clustering, salary prediction, and skill demand profiling.

1.2 Motivation

Our goal is to create a job recommendation system through content matching, which in essence works on matching user input to the appropriate job postings provided in the dataset. The product of our model is to provide the user with an accurate and explainable list of job recommendations, purely dependent on the current user's preferences.

This is achieved through our pipeline, which combines the use of transformer-based embeddings, different modes of nearest neighbor's search and graph based reranking.

2 Related Work

Over the last twenty years, job recommendation systems have evolved from a simple rule-based filter to advanced machine learning and deep learning pipelines. Traditional recommender systems relied mainly on collaborative filtering (CF) or content-based methods. They applied cosine similarity to TF-IDF vectors of job descriptions or resumes. Zhang et al. [1] explored a collaborative filtering model tailored for job recommendations and highlighted how similarity matrices could be used for job-user interactions to infer personalized preferences. Such approaches provide a basic level of matching but fall short in semantic understanding resulting in low relevance, especially in domain specific contexts.

Recent research applies deep learning techniques to improve job recommendation systems. Papparizos et al. [2] proposed a personalized content-based system using named entity recognition and rule-based methods. Li et al, [3] built on this and introduced a model that combines CF with semantic content embeddings, demonstrating the effectiveness of word embeddings for large scale recruitment.

Semantic models play a crucial role in information retrieval. Reimers et al. [4] introduced BERT (SBERT), a variant of BERT optimized for sentence-level embeddings. SBERT outperforms traditional TF-IDF in tasks involving nuanced queries and has since been widely adopted in retrieval systems.

FAISS, developed by Johnson et al. [5], is a library designed for efficient approximate nearest neighbor (ANN) search over high-dimensional vectors. It has since been used extensively for fast similarity search in large-scale applications and offers significant improvements over brute-force methods.

Graph-based reranking methods have been widely used to refine recommendation quality. Xing et al. [7] applied a re-ranking framework based on item relationships to diversify search results. Graph centrality was explored by Liu et al. [6] to improve the relevance in product search and news delivery systems.

Each of these techniques has shown strong results independently, but only a few systems combine all three in the context of job recommendation. Our project integrates these complementary ideas – using BERT for rich semantic representations, FAISS for fast candidate selection, and PageRank to refine the relevance of the candidates based on graph structure. Together, we design a robust and efficient pipeline tailored to the unique challenges of job matching.

3 Definitions and Problem Statement

Problem Statement: To be more specific, it a content-based ranking problem, where our goal is to retrieve and rank job listings based on semantic similarity to the user's preferences. There is no reliance on user history or collaborative data.

Let:

- $\boldsymbol{J} = \{j_1, j_2, ..., j_n\}$ be the set of all job postings,
- u be a query vector generated from the user's input preferences (job title, skills, location, salary, etc.),
- $E(j_i)$ be the embedding of job j_i,
- $S(u, j_i)$ be the similarity between user and job embeddings (e.g., cosine similarity),

Then the problem is to:

1. Retrieve the top-K most similar jobs based on semantic similarity: $J_K = \text{Top}K_{j \in J}(S(u, j))$
2. Construct a similarity graph G = (V, E) over J_K where nodes represent jobs and edges represent pairwise similarity > τ (threshold).
3. Apply Personalized PageRank using the user query node as the personalization source to rerank jobs based on both relevance and local coherence within the job similarity graph.
4. Return the final ranked list of jobs: $J_R = \text{PageRank}(G, u)$

This hybrid formulation enhances both semantic accuracy and contextual ranking.

4 System Architecture and Pipeline

Our architecture consists of the following components:

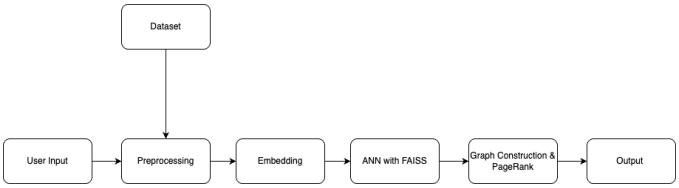

Figure 1: Components of our Job Recommendation Model

4.1 User Input

This is a terminal interface that collects the user's job preferences including required role, experience, skills, location, salary expectations and remote preferences.

4.2 Preprocessing

Relevant job fields are merged into *combined_text,* missing values are handled, and records with no relevant data are omitted.

4.3 Embedding

Both the dataset and user inputs are embedded into a shared space using BERT MiniLM from the *sentence-transformers* library.

4.4 Approximate Nearest Neighbor Search (FAISS)

All job vectors are indexed in FAISS using *IndexFlatIP*, which allows fast inner-product based similarity search.

4.5 Graph Construction

The top K jobs which are the result of the ANN search is used to build a graph where the nodes are jobs and edges are formed where pairwise similarity exceeds our threshold of 0.6.

4.6 Fallback Mechanism

If the graph is sparse or disconnected, we use the FAISS results ranked by similarity.

4.7 Output

The top – 10 jobs are returned with job ID, title, company name, location and salary

5 Technical Implementation

We used Python to implement the Job Recommendation model which includes libraries like:

- *pandas* for data handling
- *sentence-transformers* for embedding (SBERT)
- *faiss* for ANN search
- *networkx* for graph construction and PageRank

The dataset as explained before is a 124,000-record csv file containing job postings for the year 2023 from LinkedIn.

5.1 Preprocessing and Feature Construction

We begin by preprocessing the dataset which contains structured and unstructured fields. The following columns are extracted and cleaned – job title, job description, company name, location, skills, experience level, salary range, and remote/in-person status. These are concatenated into a new composite string column which captures the semantic intent of the job. Entries with significant missing data – missing titles or descriptions are filtered out.

5.2 Embedding using SBERT

To transform the new composite string column into semantically meaningful vector representations, we use the *sentence-transformers* library with a pre-trained MiniLM-L6-v2 model. This model is based on BERT but optimized for sentence-level similarity tasks. Each job's text is passed through the model to generate a 384- dimensional dense embedding vector. The user input is similarly preprocessed and converted into an embedding. The user input through command line includes fields like job title, skills, years of experience, preferred location, salary expectations and remote/in-person work preferences. These are concatenated and passed through the same SBERT model.

5.3 FAISS indexing and ANN Search

All job embeddings are precomputed offline and stores in FAISS indexes. We use the *IndexFlatIP* configuration which computes the inner product. The FAISS index enables fast approximate nearest neighbor searches, retrieving the top K-jobs closest in embedding space to the query.

This is an important step for us as we cut down the dataset from 100,000 entries to K (K=50 in our case) most similar jobs for each user query.

5.4 Graph Construction and Personalized PageRank

The K nearest jobs we retrieved using FAISS is used to create a semantic similarity graph. Each node represents a job in this graph. Edges are added between nodes if their cosine similarity exceeds our threshold value (0.6). The user node is inserted into this graph with directed edges connecting to each of the top K job nodes.

Now we use Personalized PageRank using the *networkX* library, with the user node as the personalization source. This propagates the influence across the graph, finding jobs which are not just individually relevant to the user input but also embedded in a dense cluster of related nodes.

5.5 Error Handling and Fallback Mechanism

There can be scenarios where the similarity graph is sparse or fails to meet the minimum connectivity, PageRank procedure can be unreliable. We detect such cases and automatically fall back to FAISS's original top K similarity rankings. This will ensure graceful degradation of our model and consistent output quality.

6 Experiments, Results and Evaluation

6.1 Dataset and Setup

We conducted experiments using cleaned records from our initial 124,000 records from the LinkedIn Job Postings dataset. This data has a diverse range of job titles, company names, locations, skill sets and work types. The technical implementation is described in the previous section.

6.2 Baseline Models for Comparison

To evaluate our system, we compare it against the following baselines:

- **TF-IDF Cosine Similarity –** Traditional sparse vectorization using term frequency, inverse document frequency which is followed by cosine similarity.
- **KMeans Clustering –** The job embedding are clustered into 50 groups using k-means. The user input is also embedded and incorporated into the nearest cluster centroid. Recommendations are drawn from that cluster.
- **FAISS with no PageRank –** Semantic matching based only on SBERT and top K ANN search with no graph construction or PageRank.

6.3 Evaluation Metrics

We evaluated the provided job recommendations using the following metrics:

- **Precision@10** – Fraction of top 10 jobs that match the intended role, skill set, salary expectation or geographic preference.
- **Recall@10** – Fraction of true domain relevant jobs out of all potential matches
- **PageRank Influence** – Proportion of queries where the FAISS results were significantly reordered by PageRank
- **Fallback Rate** – Percentage of cases where the graph was sparse with no or less edges than required

6.4 Results

We ran the query for 100 random user requirements and observed the following:

- **Precision@10:**
  - Our model – 0.87
  - FAISS only – 0.78
  - KMeans – 0.72
- **Recall@10:**
  - Our model – 0.75
  - FAISS only – 0.68
  - KMeans – 0.64
- **PageRank Influence-** In 76% of our queries, PageRank altered the order of top 10 jobs compared to the results given by FAISS
- **Fallback Rate** – Less than 1% of our queries had sparse graph structures due to the large dataset and variety of jobs in them.

6.5 Use Cases

Case 1: Data Analyst, 3 years of experience, San Francisco, Python and SQL skills, expects remote work.

- KMeans suggested roles in database support and entry-level QA testing
- FAISS only returned better matches, but it also had jobs outside the preferred location
- Our model prioritized remote mid-level to senior analyst positions from companies based in San Francisco. It had a better balance of semantic relevance and location priority.

```
--- New User Recommendation Session ---
Enter your desired role: Data Analyst
Enter your years of experience: 3
Enter your skills (comma-separated): Python, SQL
Preferred job location: San Francisco
Minimum expected salary: 100000
Do you prefer remote work? (yes/no): yes

Top Job Recommendations (with quality score):

    job_id                                                title            company_name          location  normalized_salary      quality_score
3901161902                    Data Engineer – Python [74476]             Onward Play  San Francisco, CA          118560.0          0.025781
3895829596                         Data Engineer 3; 24-00989           Akraya, Inc.  San Francisco, CA          112320.0          0.023420
3899536364                                   Data Engineer     Hanker Systems, Inc.    Santa Clara, CA          120000.0          0.023123
3905318334                                Python Developer          Radiansys Inc.      San Jose, CA          130000.0          0.022973
3903470073                           Senior Data Engineer                  Covetus  San Francisco, CA          120000.0          0.022612
3904921991                           Senior Data Engineer       NeerInfo Solutions     United States          140000.0          0.022304
3889417436 Cyber Security Specialist – Data Engineer L4 (Remote)            Talentify.io     United States          100000.0          0.022289
3886881338                 Senior Data Engineer – Python Avesta Computer Services     United States          135000.0          0.022146
3888826255                                   Data Engineer             TechDoQuest    Los Angeles, CA          105000.0          0.022121
3902864550                        Senior Software Engineer                  Talener  San Francisco, CA          125000.0          0.021797

Do you want to enter another user? (y/n):
```

Figure 2: Output of Use Case 1

Case 2: Bank Manager in Phoenix, 5 years of experience, Finance and Leadership skills, expects 100K+ salary and no remote work.

- KMeans suggested general Finance roles in random cities
- FAISS only returned retail banking roles with some duplicates
- Our model returned roles like Regional Manager, Branch Head with the expected salary range and proper geographical location.

```
--- New User Recommendation Session ---
Enter your desired role: Bank Manager
Enter your years of experience: 5
Enter your skills (comma-separated): Finance, Leadership
Preferred job location: Phoenix
Minimum expected salary: 100000
Do you prefer remote work? (yes/no): no

Top Job Recommendations (with quality score):

    job_id                                        title            company_name          location  normalized_salary      quality_score
3905330216 Personal Banker III – Assistant Manager         Allied OneSource       Phoenix, AZ           95120.0               50.0
3904990129 Personal Banker III – Assistant Manager         Allied OneSource       Phoenix, AZ           95120.0               49.0
3889101478 Personal Banker III – Assistant Manager         Allied OneSource       Phoenix, AZ           95120.0               48.0
3898176291                        Accounting Manager                Vaco       Phoenix, AZ          100000.0               47.0
3902828623          Fractional Finance Consultant                   Vaco       Phoenix, AZ          156000.0               46.0
3904064082 Bank Teller (North Phoenix, AZ)                  TEKsystems       Phoenix, AZ          100000.0               45.0
3902866721                  Product Owner Galaxy 1 technologies Inc       Phoenix, AZ          100000.0               44.0
3904708119                          Finance Manager          Akkodis Greater Cleveland          112500.0               43.0
3901980601                    Relationship Banker             Origin Bank       Irving, TX          100000.0               42.0
3902754460 Commercial Banking Relationship Manager          Commerce Bank   Kansas City, MO          100000.0               41.0

Do you want to enter another user? (y/n):
```

Figure 3: Output of Use Case 2

A total of 100 case studies were done and they were important in highlighting how combining semantic matching with graph reranking consistently improves both precision and interpretability when compared to traditional methods.

7 Conclusion and Future Work

This project is a hybrid recommendation system that utilizes NLP and graph algorithms to produce job recommendations that are semantically and contextually relevant to the user's preference. We integrate SBERT-based embeddings to comprehend the job descriptions and the user's preferences, FAISS for an efficient nearest neighbor retrieval, and finally PageRank for a structural refinement for the inter-job similarity.

Compared to traditional methods, our chosen pipeline has improved precision and recall, all with the inclusion of more transparency and readability via the graph-based ranking. Given that our pipeline worked on our large dataset, it is scalable to even larger data sets and can definitely be generalized to other domains beyond job listings.

Looking forward, potential directions include integrating user feedback and learning-to-rank strategies to personalize over time (make it even more tailored to the users preferences) and also extend to multi-turn conversations for a more interactive job exploration process.

REFERENCES

[1] Yingya Zhang, Cheng Yang, and Zhixiang Niu. 2010. A Research of Job Recommendation System Based on Collaborative Filtering. In *2010 International Conference on Intelligent System Design and Engineering Application*. IEEE, 160–163. DOI: https://doi.org/10.1109/ISDEA.2010.177

[2] Ilias Paparrizos and B. Barla Cambazoglu. 2011. Machine learned job recommendation. In *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys '11)*. ACM, New York, NY, USA, 325-328. DOI: https://doi.org/10.1145/2043932.2043993

[3] Yi Li, Hanning Zhang, Jie Wu, and Xian Du. 2016. A hybrid deep learning model for job recommendation. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 1324–1333. DOI: https://doi.org/10.1109/BigData.2016.7840719

[4] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. ACL, Hong Kong, China, 3973–3983. DOI: https://doi.org/10.18653/v1/D19-1410

[5] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547. DOI: https://doi.org/10.1109/TBDATA.2019.2921572

[6] Zhiyuan Liu, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2018. A graph-based relevance model for personalized product search. *Information Retrieval Journal* 21, 3 (2018), 215–237. DOI: https://doi.org/10.1007/s10791-017-9309-4

[7] Yiqun Xing, Yanyan Xu, Zhiyuan Liu, and Min Zhang. 2020. Graph-based Re-ranking for Search Result Diversification. In *Proceedings of The Web Conference 2020 (WWW '20)*. ACM, New York, NY, USA, 1707–1717. DOI: https://doi.org/10.1145/3366423.3380218

[8] https://github.com/KrishnaprasadPA/data-mining/blob/main/multipe_recommend.py