ResuMatch: Smart Resume Parser & Job Fit Analyzer

Naga Pavithra Lagisetty
Khoury School of Computer Sciences
Northeastern University
Boston, MA, 02115
lagisetty.n@northeastern.edu

Hithaishi Raghavendra Reddy
Khoury School of Computer Sciences
Northeastern University
Boston, MA, 02115
raghavendrareddy.h@northeastern.edu

Vinisha Sunkara
Khoury School of Computer Sciences
Northeastern University
Boston, MA, 02115
sunkara.vi@northeastern.edu

Sharan Giri
Khoury School of Computer Sciences
Northeastern University
Boston, MA, 02115
giri.sha@northeastern.edu

Abstract

We developed ResuMatch, a machine learning-based system that analyzes resumes and recommends relevant job roles. The project applies natural language processing (NLP) techniques such as TF-IDF vectorization and dense text embeddings to convert raw resume data into meaningful numerical representations. A One-vs-Rest logistic regression model, combined with multi-label binarization and probability calibration, is used to predict the most suitable job titles for each candidate. The system also highlights matched and missing skills based on role-specific keyword patterns. Experimental results on the current dataset show high predictive performance and actionable feedback for job seekers. Future iterations can benefit from larger datasets and more advanced embeddings for improved role alignment.

1. Introduction

Finding the right job for the right candidate is a core challenge in today's recruitment ecosystem. The Resume-Job Matching System uses machine learning techniques to analyze resumes and recommend relevant job roles based on the content of the resumes and job descriptions. This task involves mapping a resume's text and a job description into a matching score that quantifies the alignment between a candidate's qualifications and a job's requirements. In this system, we essentially learn to map resume content, represented as a multi-dimensional feature vector, to a job title that best matches the applicant's profile. The process is broken down into two main components: feature extraction and classification. Feature extraction involves processing the text of the resume and job description to generate meaningful representations. We use TF-IDF (Term Frequency-Inverse Document Frequency) for feature extraction, where each word in the resume and job description is transformed into a numerical vector representation based on its frequency and relevance. This allows the system to capture the most important words and phrases that describe a candidate's skills and qualifications. For the classification task, we use a One-vs-Rest logistic regression model with MultiLabelBinarizer, enabling the system to handle resumes that map to multiple job roles. To improve the reliability of prediction scores, CalibratedClassifierCV is used for probability calibration. In addition to job role prediction, the system provides users with insights into matched and missing skills, guiding them toward upskilling opportunities. Evaluation on a labeled dataset demonstrates strong predictive performance, with future enhancements aimed at expanding the dataset and incorporating richer contextual features. To assess performance, the system evaluates the accuracy of its predictions and the quality of job recommendations. The results suggest that the model performs well on the given dataset, providing relevant and accurate job role suggestions. Further data may improve the system's expressiveness and lead to better predictions.

2. Methodology

We implemented the Resume-Job Matching System using machine learning techniques to match resumes with job roles, following an architecture inspired by previous work in the domain [1]. Some architectural decisions were based on similar models, while others were made based on our specific project needs. Consequently, the results and implementation may slightly differ from any previous work, but the core principles remain consistent.

2.1 Dataset and Preprocessing:

The dataset [2] used in this project has been curated and processed by Neural Frame AI obtained with consent. It includes structured information on career objectives, skills, education, work experience, certifications, and other relevant details, making it ideal for job matching, candidate profiling, and resume parsing, making it ideal for job matching, candidate profiling, and resume parsing. Since the resume data is diverse, unstandardized and contains a mix of structured lists / free-form text, missing values, extensive preprocessing skills were performed to handle inconsistencies in the dataset. Key steps include :

1. Missing Value Analysis, Field Pruning & Nested Lists Handling : Null value percentage was calculated for every field. Fields with consistently high missing rates were dropped.Converted fields that were lists stored as strings, into actual Python lists using ast.literal_eval and cleaned them to remove placeholder values such as None, 'N/A', and empty strings.

2. Date Parsing & Experience Calculation: Employment date fields (start_dates, end_dates) were parsed into datetime objects using dateutil.parser. Strings such as "present"/"ongoing"/"current" were converted to current date. Total work experience was calculated by taking the difference between the earliest start date and the latest end date.

3. Educational History Parsing: The parsing_years field contained multiple years and noise ("Expected May 2020"). A custom function was used to extract valid years from this field and identify the index of the most recent graduation year. This index was used to extract the corresponding latest educational results of the candidate. Diverse result formats (e.g., "3.5/4.0", "87%", "gold medalist", "cum laude") were parsed and normalized to a GPA scale (0.0 - 4.0). A dedication function was defined to handle the numerical and qualitative grading pattern.

4. Merging Candidate Skill-sets and Professional Responsibilities: The newline character was used as a delimiter to create a structured list of responsibilities for "responsibilities" and "responsibilities.1" fields and merged into a single column "combined_responsibilities". The fields "skills", "related_skills_in_job", and flattened "certification_skills" are unified and stored together to capture all unique skills picked up by candidates throughout their educational and professional journey.

This comprehensive preprocessing pipeline ensures the dataset is clean, consistent, and ready for extensive feature extraction and model training through the preserved semantic richness and reduced noise and redundancy. This helps to significantly improve the ability of downstream models to learn patterns from the data in a much better manner, while improving scalability with reduced fields list.

2.2 Method:

The ResuMatch system applies a structured machine learning pipeline to analyze resumes and recommend relevant job roles. The process begins with data preprocessing, where raw text from resumes is cleaned and standardized. This involves converting all text to lowercase, removing punctuation, eliminating stop words, and tokenizing the text into individual terms. These steps help reduce noise and ensure consistency across the dataset. Once preprocessed, the textual data is transformed into numerical features using TF-IDF vectorization. TF-IDF captures the importance of words by measuring their frequency relative to the entire corpus, allowing the system to focus on terms that are more informative for classification. These vectorized features are then used as input to the machine learning model. To further evaluate how well a resume aligns with a specific job description, cosine similarity is applied to the TF-IDF vectors of both texts. This measures the angular similarity between the two, producing a match percentage that reflects how closely the resume matches the job requirements. These vectorized features and similarity scores are then used as input to the machine learning model.

Given that a single resume can correspond to multiple job roles, the problem is framed as a multi-label classification task. To handle this, job titles are encoded using a MultiLabelBinarizer, which converts the target labels into binary vectors. We use a One-vs-Rest logistic regression model for classification, training a separate binary classifier for each job title. This model is well-suited for high-dimensional text data and supports multi-label prediction efficiently. Additionally, we use CalibratedClassifierCV to calibrate the model's output probabilities, ensuring that the confidence scores for each predicted job role are reliable and interpretable. To enhance the system's feedback mechanism, we implemented a skill matching module that provides insights into matched and missing skills for each predicted job title. For each role, we maintain a predefined list of key skills. The system checks which of these skills are present in the resume, highlighting them as "matched," while the absent ones are listed as "missing." This offers valuable guidance for job seekers looking to tailor their resumes for specific roles.

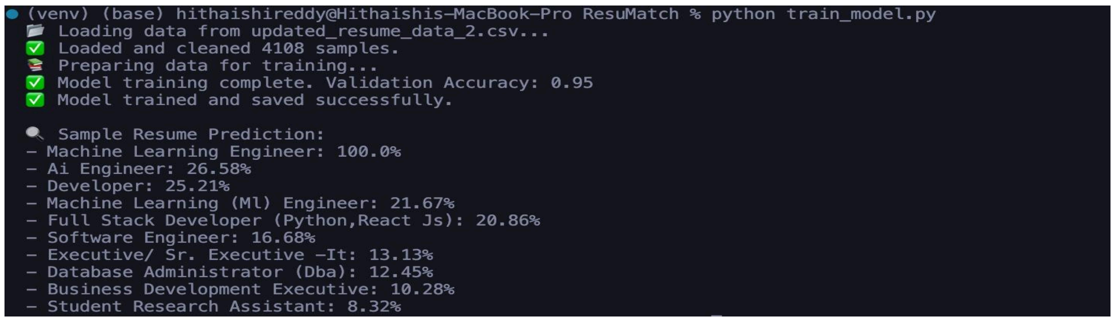
The model's performance is evaluated using standard multi-label metrics such as accuracy, Hamming loss, precision, recall, and F1-score. Our experiments showed a marked improvement in performance after applying preprocessing and feature engineering, with accuracy improving from approximately 86% to 95%.

The website connects to a back-end built with Flask, which calculates match scores and suggests improvements for resumes. It provides users with real-time insights to enhance their resumes as they apply for jobs, ensuring better alignment with job requirements.

2.3 Training:

In the project, the dataset is split into training and testing sets using the train_test_split function from scikit-learn. TF-IDF Vectorization is then applied to convert the text into numerical features, capturing the importance of each word while ignoring common stopwords. A One-vs-Rest (OvR) multi-label classification approach is used, where Logistic Regression serves as the base classifier. The model is trained using Logistic Regression with a balanced class weight to handle any imbalances in the dataset and CalibratedClassifierCV to calibrate the classifier and improve probability predictions. The split ratio used in the python file is 80% for training and 20% for testing. The validation accuracy is calculated to measure how well the model predicts job titles for resumes it hasn't encountered before. Since the task is multi-label classification, precision, recall, and F1-score are also used to evaluate performance for each job title, considering how well the model identifies relevant job titles and avoids false positives or negatives.

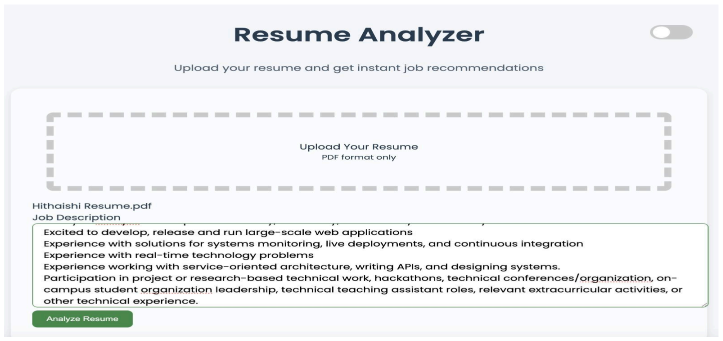
3. Results



```
● (venv) (base) hithaishireddy@Hithaishis-MacBook-Pro ResuMatch % python train_model.py
  📂 Loading data from updated_resume_data_2.csv...
  ✅ Loaded and cleaned 4108 samples.
  📚 Preparing data for training...
  ✅ Model training complete. Validation Accuracy: 0.95
  ✅ Model trained and saved successfully.

  🔍 Sample Resume Prediction:
  - Machine Learning Engineer: 100.0%
  - Ai Engineer: 26.58%
  - Developer: 25.21%
  - Machine Learning (Ml) Engineer: 21.67%
  - Full Stack Developer (Python,React Js): 20.86%
  - Software Engineer: 16.68%
  - Executive/ Sr. Executive -It: 13.13%
  - Database Administrator (Dba): 12.45%
  - Business Development Executive: 10.28%
  - Student Research Assistant: 8.32%
```

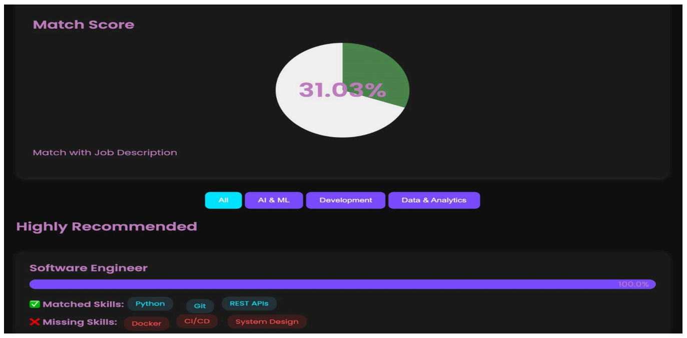
Figure 1: Validation Accuracy and Sample Prediction. The image shows the output from training the job prediction model using our dataset. A total of 4108 resume samples were successfully loaded, cleaned, and used to train the model. The model achieved a high validation accuracy of **95%**, indicating strong predictive performance.



Hithaishi Resume.pdf

Job Description

Excited to develop, release and run large-scale web applications
Experience with solutions for systems monitoring, live deployments, and continuous integration
Experience with real-time technology problems
Experience working with service-oriented architecture, writing APIs, and designing systems.
Participation in project or research-based technical work, hackathons, technical conferences/organization, on-campus student organization leadership, technical teaching assistant roles, relevant extracurricular activities, or other technical experience.



✅ Matched Skills: Python Git REST APIs

❌ Missing Skills: Docker CI/CD System Design

Figures 2 & 3: Web Interface and Resume Match Score. Users can upload their resume in PDF format and input a job description to receive tailored job recommendations. Once the user clicks "Analyze Resume", the system processes the input for further analysis.

4. Discussion

The project aims to develop a system that automatically matches resumes to job descriptions and provides role recommendations. The model successfully predicts job titles and identifies missing skills from resumes. However, it faces challenges, primarily due to the quality and diversity of the training data. Since the dataset is limited, the model struggles with resumes containing underrepresented or unique job titles and skills, and may not perform well on out-of-domain resumes that differ from the training examples.

Our model performs well in predicting job titles based on validation accuracy and precision/recall scores. However, when evaluating with advanced metrics like multi-label classification accuracy, we found that the model tends to match resumes with a broad range of job titles, reducing specificity in recommendations. While we did not compare directly with other models, we believe that improvements in data diversity and model tuning could enhance its performance.

Increasing the training data improved the model's performance, enabling it to capture a wider range of skills and job titles. While the model's performance plateaued with the current dataset, we believe adding more diverse data could further enhance it. In the future, we plan to refine the model with larger datasets, fine-tune it for specific industries, and conduct qualitative evaluations to understand how it makes predictions. This could help improve the model's handling of unique job titles and skills. Additionally, the model could be extended for use in other datasets or as a pre-trained solution for specific industries.

5. Conclusion

In conclusion, the ResuMatch: Smart Resume Parser & Job Fit Analyzer project successfully builds a system that matches resumes to job descriptions and provides role recommendations based on resume content. The model effectively predicts job titles and identifies missing skills, but it faces challenges with unique or out-of-domain job titles and resumes that differ from the training data. These limitations suggest that the model's performance can be further improved with a more diverse dataset and advanced evaluation techniques.

Future work will focus on expanding the training data to include a wider variety of resumes and job titles and experimenting with deep learning [3] and attention mechanisms [4, 5], to improve the model's accuracy and specificity in job role recommendations. Additionally, fine-tuning the model for specific industries and conducting qualitative evaluations will offer deeper insights into the model's decision-making process, ultimately leading to a more robust system capable of handling diverse and complex resume-job matching tasks.

6. Acknowledgement

This project was completed for CS6120 Natural Language Processing under the instruction of Professor Uzair Ahmad.

References:

[1] S. Pabalkar, P. Patel, R. Choudhary, V. Panoch, S. Yadav and H. Ghogale, "Resume Analyzer Using Natural Language Processing (NLP)," *2024 International Conference on Intelligent Systems and Advanced Applications (ICISAA)*, Pune, India, 2024, pp. 1-6, doi: 10.1109/ICISAA62385.2024.10828940.

[2] Resume Dataset, Retrieved February 2025, from Kaggle : Resume Dataset

[3] Zhang, Z., Luo, Y., Wen, Y., & Zhang, X. (2022, August). Cycleresume: A cycle learning framework with hybrid attention for fine-grained talent-job fit. In *CAAI International Conference on Artificial Intelligence* (pp. 260-271). Cham: Springer Nature Switzerland.

[4] Wang, Z., Wei, W., Xu, C., Xu, J., & Mao, X. L. (2022). Person-job fit estimation from candidate profile and related recruitment history with co-attention neural networks. *Neurocomputing*, *501*, 14-24.

[5] Huang, Y., Liu, D. R., & Lee, S. J. (2023). Talent recommendation based on attentive deep neural network and implicit relationships of resumes. *Information Processing & Management*, *60*(4), 103357.