SmartApply: Automated LinkedIn Messaging for Job Hunters

Poonam Kishor Pawar
Department of Engineering
Applied Machine Learning 2

***Abstract*—Job hunting for data-focused roles often involves tedious manual tasks such as scanning job postings, tailoring resumes, and drafting personalized messages. SmartApply aims to automate this process by intelligently matching candidate resumes with job postings and generating customized LinkedIn messages. This paper presents a data-driven approach using a Kaggle job posting dataset to simulate the workflow. The system analyzes job descriptions, extracts required skills, compares them with candidate profiles, and generates a personalized outreach message highlighting the most relevant qualifications. Early implementation demonstrates the effectiveness of text preprocessing, similarity-based matching, and template-driven natural language generation. Initial evaluation suggests promising accuracy in skill matching and message relevance, with ongoing improvements focused on semantic understanding and interface usability.**

***Index Terms*—Job–Resume Matching, Personalized Message Generation, Natural Language Processing, Cosine Similarity, TF-IDF, Data Scientist, Machine Learning Engineer, AI Engineer**

I. Introduction

Finding a suitable role in data science, machine learning, or AI often requires extensive customization of resumes and messages for each job posting. Manual approaches are inefficient and prone to oversight. With the rise of online platforms such as LinkedIn, automated tools can significantly reduce the time and effort needed for personalized job outreach.

SmartApply leverages natural language processing (NLP) techniques to match candidate skills with job requirements and generate relevant messages. Using a curated Kaggle dataset of job postings, the system focuses on roles including Data Scientist, Machine Learning Engineer, and AI Engineer.

The primary goal is to develop a robust pipeline for resume–job matching and automatic message generation, enabling candidates to streamline job applications while maintaining personalization.

II. Motivation

The demand for data-focused talent is high, yet the process of applying remains largely manual. SmartApply is motivated by the need to:

Automate resume–job matching for better efficiency.

Reduce manual effort in crafting personalized messages.

Highlight candidate skills effectively, improving job application success rates.

Provide a reproducible pipeline for experimentation and future enhancements.

Automated matching can also help recruiters identify qualified candidates faster, while enabling applicants to reach out more intelligently to relevant roles.

III. Dataset And Preprocessing

A. Dataset Description

The dataset utilized in this study comprises a collection of job postings and candidate resumes focused on data-centric roles such as Data Scientist, Machine Learning Engineer, and AI Engineer. The job dataset includes fields such as job title, company name, description, skills, experience level, work type, location, and salary information. The resume dataset contains text-based profiles with fields like education, experience, skills, and projects, extracted from uploaded documents or user-provided text. This dataset forms the foundation for the job–resume matching system by enabling semantic similarity analysis between candidate profiles and job requirements.

B. Data Preprocessing

To ensure consistency and enhance the performance of downstream models, several preprocessing steps were carried out:

- **Data Filtering:** Only job titles containing Data Scientist, Machine Learning Engineer, or AI Engineer were retained for analysis.
- **Text Cleaning:** Both job descriptions and resumes were converted to lowercase, punctuation and special characters were removed, and stopwords were filtered out.
- **Tokenization & Lemmatization:** Text data was tokenized and lemmatized to standardize different word forms and improve representation quality.
- **Encoding:** The cleaned job and resume texts were transformed into dense embeddings using the SentenceTransformer model (all-MiniLM-L6-v2) to capture semantic meaning.
- **Normalization:** Embedding vectors were normalized to ensure consistent magnitude before similarity computation.
- **Visualization:** Exploratory analysis, including word frequency plots and correlation heatmaps, was conducted to verify data distribution and identify common skills and terms across job postings.

IV. Exploratory Data Analysis

A. Summary of findings

- **Job Role Distribution:** The dataset primarily focuses on data-related roles such as Data Scientist, Machine Learning Engineer, and AI Engineer, with a balanced distribution across these categories.
- **Text Length Variation:** Analysis of job descriptions and resumes revealed that job descriptions tend to be longer on average, providing detailed requirements, while resumes are more concise, emphasizing skills and experience.
- **Common Skills:**Frequently occurring skills identified include Python, Machine Learning, SQL, TensorFlow, and Tableau, aligning well with industry demands in data-driven roles.
- **Experience & Salary Correlation:** Preliminary analysis indicated a positive correlation between years of experience and normalized salary values across job listings.
- **Missing and Noisy Data:** Some missing values were found in columns such as normalized_salary and skills_desc, which were handled during preprocessing to ensure data completeness.

B. Visualizations

Figure 1 shows the top 10 locations.

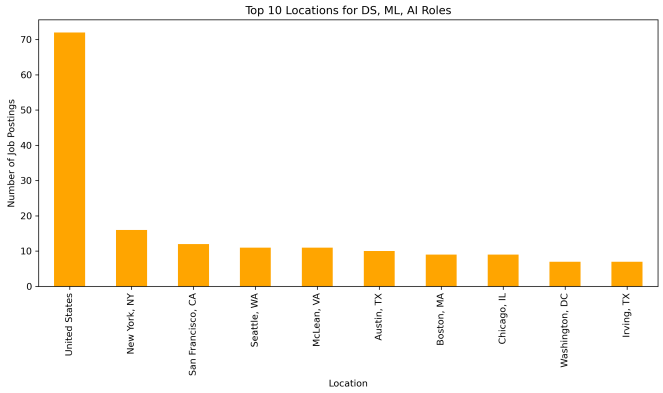

Fig. 1. Top 10 location for DS, ML, AI roles

Figure 2 shows the Salary by Experience Level.

Figure 3 shows a Salary by workType

V. System Architechture

A. Overview

The SmartApply system is designed with a multi-layered architecture to facilitate automated resume-job matching and message generation. The architecture consists of three primary layers: the Presentation Layer, the Application & Business Logic Layer, and the Data Layer. This modular design ensures scalability, maintainability, and efficient execution of AI-driven processes.

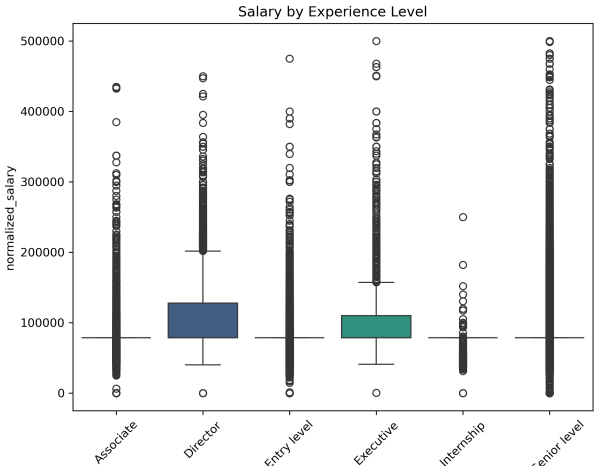

Fig. 2. Salary by Experience Level

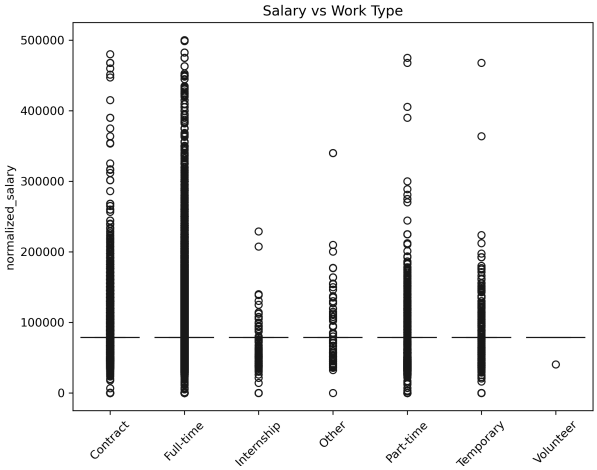

Fig. 3. Salary by workType

B. Layer-wise Architecture

1. Presentation Layer

Component: SmartApply UI
Functionality:

- Allows users to upload resumes in .txt formats.
- View matched job listings with similarity scores.
- Receive generated LinkedIn message drafts.

Interaction: User actions in the UI trigger API calls to the backend services.

2. Application & Business Logic Layer

Component: Job Matcher API Service

- Parses and cleanses the uploaded resume.
- Fetches job data and precomputed embeddings.
- Calculates semantic similarity between resume and job postings.
- Generates draft messages based on matched jobs.

- Stores and returns results to the UI.

Component: ML/AI Microservices

- **Resume Parser:** Uses SpaCy/NLTK for tokenization, lemmatization, and entity extraction.
- **Embedding Service:** Leverages Sentence-BERT to generate dense vector representations for resumes and job postings.
- **Message Generator:** Employs generative models (GPT/T5/OpenAI) to create personalized LinkedIn messages.

3. Data Layer

Components:

- **Job Database:** Stores job postings in formats such as CSV, JSON, or SQL databases.
- **Embeddings Cache:** Maintains precomputed embeddings for faster similarity computation (using Redis or Pickle).

Figure 4 Architechture Diagram.

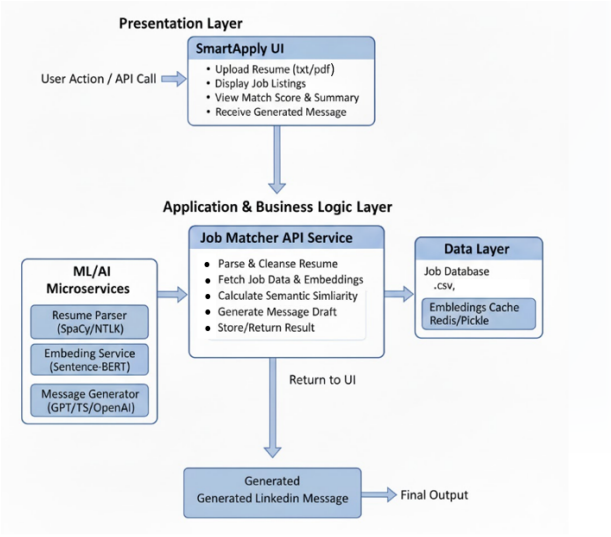

Fig. 4. Architechture Diagram

VI. Model Implementation

A. Models Explored

SmartApply leverages multiple NLP models to achieve end-to-end automation. Sentence embeddings are computed using SentenceTransformer ('all-MiniLM-L6-v2'), summarization is performed with Facebook's BART-large-CNN, and personalized messages are generated using Google's FLAN-T5-Large. The system runs on a local CPU environment with batch-encoded embeddings and similarity computation through cosine similarity from the SentenceTransformers library. Reproducibility is ensured by modularized code components in job_matcher.py and app.py.

VII. Interface Prototype

The Streamlit-based web interface allows users to upload a resume (.txt) and automatically view the top-ranked job postings along with AI-generated LinkedIn messages. Inputs include a text file path to the resume, while outputs display ranked matches with similarity scores and personalized recruiter messages. The interface supports real-time interaction and generates messages under 120 words. Current limitations include model latency and lack of recruiter name personalization.

VIII. Early Evaluation and Results

Preliminary evaluation was conducted using a subset of Kaggle job postings focused on Data Scientist and ML Engineer roles. Performance was analyzed based on cosine similarity ranking accuracy and EDA correlation metrics. Figure 2 shows the numeric correlation heatmap between key salary and engagement variables, confirming strong consistency between min, max, and average salary features. This insight validates the data preprocessing and normalization pipeline used in model training

Figure 5 correlation heatmap.

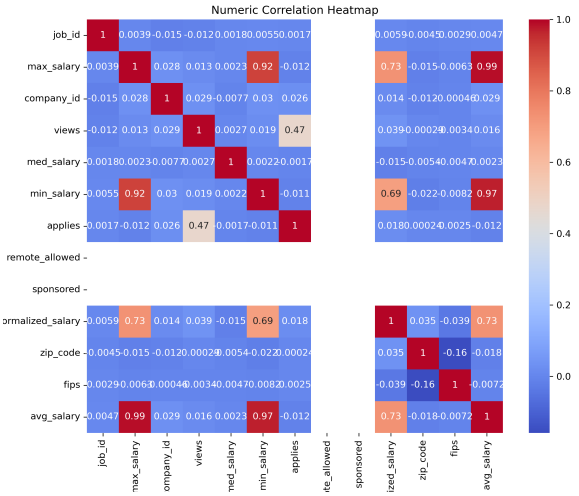

Fig. 5. correlation heatmap

IX. Challenges and Next Steps

Key challenges encountered include (1) computational limitations when embedding large datasets, (2) text truncation during summarization, and (3) message tone variability from the generative model. The next steps will focus on optimizing embedding computation with GPU acceleration, refining hyperparameters for FLAN-T5 message coherence, and improving UI responsiveness through caching. Additionally, future versions will include evaluation metrics such as semantic similarity precision and recall.

X. Responsible AI Reflection

SmartApply emphasizes responsible AI practices by avoiding bias amplification in text generation. All user resumes are processed locally without storage to preserve privacy. Generated recruiter messages are filtered to maintain professionalism and non-discriminatory language. Future improvements include integrating fairness checks for candidate-job matching scores and ensuring transparency in automated message personalization decisions.