TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

A
PROJECT REPORT
ON

RESU-MATCH: INTELLIGENT RECRUITMENT WITH AI

SUBMITTED BY:

ASHIM SAPKOTA (PUL078BEI006)
BATSAL BHUSAL (PUL078BEI009)
BIRAT THAPA (PUL078BEI012)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

April, 2025

Copyright

Acknowledgments

We would like to express our sincere gratitude to the Department of Electronics and Computer Engineering, Pulchowk Campus, for providing us with the opportunity to propose and present our minor project. The department's continuous encouragement and support have been instrumental in guiding us.

We are deeply thankful to the Head of the Department, along with Deputy Heads of Department, for fostering an academic environment that promotes research, innovation, and growth.

We also extend our gratitude to the IC Chair Asst. Prof. Anku Jaiswal and Asst. Prof. Anila Kansakar, for their guidance and feedback, which have been crucial in shaping our project proposal.

Furthermore, we acknowledge the support of our seniors, for their advice and mentorship, which have motivated us throughout this process. We are equally grateful to our peers and colleagues for their cooperation and encouragement.

This proposal submission would not have been possible without the collaborative efforts and guidance of everyone mentioned above, and for that, we remain sincerely thankful.

Thank you.

Sincerely,

Ashim Sapkota

Batsal Bhusal

Birat Thapa

Abstract

The recruitment process often involves the challenge of efficiently reviewing and shortlisting resumes from a large pool of candidates, which can be both time-consuming and prone to human biases. To address these challenges, our system, Resu-match, leverages Natural Language Processing (NLP) techniques to automate the matching of resumes with job descriptions. The system employs a two-layer filtering mechanism to rank resumes. At the core of the system is a SBERT-based similarity ranking mechanism, which generates embeddings to evaluate the relevance of each resume to the job description. This is followed by a ML classifier, which acts as a second layer of filtering to further refine the ranking, ensuring that only the most relevant resumes are shortlisted for review. The combination of these two techniques helps improve both the speed and accuracy of candidate matching, reducing the time recruiters spend manually reviewing resumes while improving the overall quality of their hiring decisions. The platform is fast, scalable, and user-friendly, with the ability to handle large-scale recruitment tasks while minimizing bias in the hiring process. In future versions, we plan to enhance the platform with additional features for job seekers and greater transparency in the resume ranking process, ultimately providing a more comprehensive solution for both recruiters and candidates.

Keywords: *NLP, SBERT, Embeddings, ML*

Contents

List of Figures

List of Tables

List of Abbreviations

NLP	Natural Language Processing
BERT	Bidirectional Encoder Representations from Transformers
S-BERT	Sentence-BERT
MLM	Masked Language Modelling
REST API	Representational State Transfer Application Programming Interface
JWT	Json Web Tokens
MSE	Mean Squared Error
MAE	Mean Absolute Error
PDF	Portable Document Format
ATS	Application Tracking System
ANN	Artificial Neural Network
RNN	Recurrent Neural Network

1. Introduction

1.1 Background

Recruitment is a vital process in any organization, as hiring the right talent directly impacts business success. Traditional recruitment methods often involve manually reviewing resumes, shortlisting candidates, and conducting interviews. However, with the rise of digital job applications, recruiters receive thousands of resumes for a single job posting, making manual screening an inefficient and time-consuming task.

Human biases in hiring can also lead to unfair selection processes, where candidates may be evaluated subjectively rather than based on their qualifications. Moreover, keyword-based applicant tracking systems (ATS) lack the intelligence to understand the true context of a resume, resulting in the rejection of potentially qualified candidates. To address these challenges, AI-powered recruitment solutions have emerged as a game-changer in the hiring industry.

Resu-Match is an intelligent recruitment system designed to streamline the hiring process using artificial intelligence. By integrating Natural Language Processing (NLP) and Machine Learning (ML) techniques, Resu-Match can analyze resumes, extract key skills, and match candidates to job descriptions with high accuracy. The platform leverages FastAPI for efficient backend processing, PyTorch for deep learning-based resume analysis, and React for an interactive and user-friendly frontend.

1.2 Problem statements

The hiring process faces several significant challenges, which result in inefficiencies, delays, and potential biases in candidate selection. Some of the key issues include:

- **Time-Consuming Resume Screening:** Recruiters manually review hundreds or thousands of resumes, making the process slow and labor-intensive.
- **Inaccurate Candidate-Job Matching:** Traditional keyword-based systems often fail to understand the context of resumes and job descriptions, leading to irrelevant recommendations.
- **Unconscious Bias in Hiring:** Manual screening is influenced by personal biases, potentially leading to unfair hiring decisions.

- **Lack of Scalability:** Large organizations struggle to efficiently process a high volume of job applications.
- **Poor Candidate Experience:** Delayed responses and unclear job matching reduce the engagement of potential candidates.

The need for an AI-powered automated recruitment system is evident. By reducing manual effort, improving accuracy, and promoting fairness, such a system can significantly enhance the hiring process.

1.3 Objectives

The primary goal of Resu-Match is to develop an intelligent recruitment platform that automates resume screening and job matching using AI. The specific objectives of the project include:

- Design an AI-based system that can analyze resumes and job descriptions using Natural Language Processing (NLP).
- Implement a deep learning-based matching algorithm that ranks candidates based on their qualifications and a ML classifier to differentiate different job category making it a two layer holistic filtering mechanism.
- Develop a fast and scalable backend using FastAPI for real-time resume processing.
- Create an interactive and user-friendly frontend using React to enhance recruiter and candidate experience.
- Ensure bias-free hiring by using ethical AI practices to prevent discrimination in candidate selection.
- Provide detailed analytics to recruiters, including resume scores, candidate rankings, and hiring insights.

Resu-Match aims to improve the efficiency, accuracy, and fairness of the recruitment process by leveraging the latest advancements in AI and software engineering.

1.4 Scope

The project scope defines the functionalities and limitations of the Resu-Match platform. The key aspects of the system include:

- **Resume Parsing:** The system will extract structured information such as name, education, work experience, and skills from resumes.

- **AI-Powered Candidate Matching:** Using NLP, deep learning and ML classifier the system will compare resumes with job descriptions then categorize the resumes based on the resumes and also provide with similarity ranking making it a robust two layer filter mechanism.
- **Web-Based Platform:** The platform will have a user-friendly interface for recruiters to upload job descriptions and view AI-driven recommendations.
- **Real-Time Processing:** The system will process resumes quickly and return recommendations within seconds, ensuring efficiency in high-volume recruitment.
- **Integration with ATS Systems:** The platform can be made compatible with existing Applicant Tracking Systems (ATS) used by organizations.
- **Scalability and Security:** The system will be designed to handle a large number of resumes while ensuring data privacy and security.
- **Bias Reduction:** The AI model will be trained using diverse datasets to minimize discrimination and ensure fair candidate evaluation.

2. Literature Review

2.1 Related work

The application of Artificial Intelligence (AI) and Natural Language Processing (NLP) in recruitment has been extensively studied, with various approaches developed to enhance resume parsing and candidate-job matching.

One significant advancement is the use of Transformer-based models, such as BERT and GPT, in resume screening. [1] demonstrated the ability of BERT to extract contextual meaning from text, significantly improving semantic matching in unstructured resumes. Similarly, GPT-based approaches have been explored to generate and analyze job descriptions, ensuring better alignment between job roles and candidate profiles [2].

Another notable work is [3], which proposed a hybrid approach combining traditional NLP pipelines with Transformer architectures to parse job descriptions effectively. The study reported improvements in skill extraction and contextual understanding compared to rule-based systems.

[4] explored the application of machine learning models like Support Vector Machines (SVMs) and Random Forests in resume classification. While effective for structured data, these models struggled with unstructured resumes, emphasizing the need for deep learning-based approaches.

Existing AI-driven recruitment platforms, such as CiiVSOFT and Searchlight.ai, provide resume parsing and candidate matching services. However, these systems often operate as black-box models, limiting transparency and adaptability for different organizations [5].

Sieve.ai [6] is an AI-powered resume screening tool that uses traditional machine learning classifiers to evaluate and rank resumes. Unlike our approach, which leverages fine-tuned SBERT for semantic similarity, Sieve.ai relies on feature-based classification for candidate matching.

Our approach differs by leveraging a fine-tuned Sentence-BERT model to enhance resume-job description matching. Unlike previous works that rely solely on keyword matching or pre-trained embeddings, our method incorporates both contextual sentence similarity and category-based filtering to improve accuracy and efficiency in candidate ranking.

These studies collectively highlight the potential of AI in recruitment while also identifying challenges such as handling diverse resume formats, adapting to evolving job markets, and ensuring model interpretability—areas that our project aims to address.

2.2 Related theory

2.2.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) is a field of artificial intelligence that focuses on enabling computers to understand, interpret, and generate human language. NLP combines computational linguistics with machine learning and deep learning techniques to process and analyze large volumes of natural language data.

Levels of NLP

- **Phonology** – Studies the systematic arrangement of sounds in a language. It includes the semantic use of sounds to encode meaning.
- **Morphology** – Focuses on the structure of words, breaking them into morphemes (smallest meaning units). Words consist of lexical morphemes (e.g., "table") and grammatical morphemes (e.g., "-ed" in "worked"). These can be inflectional (changing grammatical properties) or derivational (altering word meaning).
- **Lexical Analysis** – Assigns meaning to individual words using Part-of-Speech (PoS) tagging. Techniques like stop-word removal, stemming (reducing words to root forms), and lemmatization (finding base words) improve text processing.
- **Syntax Analysis** – Structures sentences using grammatical rules. It groups words into phrases, phrases into clauses, and then into full sentences to establish structural dependencies.
- **Semantics** – Determines sentence meaning by processing logical structures and resolving ambiguities. It distinguishes multiple senses of words (e.g., "bark" as a dog's sound vs. tree covering).
- **Discourse Analysis** – Examines multi-sentence structures to ensure coherence. It includes anaphora resolution (identifying references) and coreference resolution (linking expressions to the same entity).
- **Pragmatics** – Interprets implied meanings beyond literal text using contextual background knowledge. It resolves ambiguity by considering linguistic and external references.

Applications of NLP

- Virtual assistants (e.g., Alexa, Siri, Google Assistant)
- Automated customer support chatbots
- Search engine optimization and ranking
- Sentiment analysis in social media and product reviews
- Language modeling for automatic content generation
- Medical and legal document analysis
- Email filtering (e.g., spam detection)
- Financial document processing and fraud detection

Challenges

Natural Language Processing (NLP) faces challenges like ambiguity at syntactic, semantic, and lexical levels, making accurate interpretation difficult. Syntactic ambiguity leads to multiple sentence structures, while semantic and lexical ambiguities arise from words with multiple meanings. Multilingual event detection also poses difficulties, as different languages require specialized NLP pipelines. Additionally, cross-lingual tasks like named entity recognition and semantic role labeling add complexity. Implementing adaptable data-centric architectures remains a challenge. Despite these issues, ongoing research continues to improve NLP systems for better language understanding and processing.

2.2.2 Deep Learning

Deep learning (DL), a branch of machine learning (ML) and artificial intelligence (AI) is nowadays considered as a core technology of today's Fourth Industrial Revolution (4IR or Industry 4.0). Due to its learning capabilities from data, DL technology originated from artificial neural network (ANN), has become a hot topic in the context of computing, and is widely applied in various application areas like healthcare, visual recognition, text analytics, cybersecurity, and many more. However, building an appropriate DL model is a challenging task, due to the dynamic nature and variations in real-world problems and data.
Deep learning follows a structured process to solve real-world problems:

1. **Problem Definition** – Define objectives, constraints, and evaluation metrics.

2. **Data Collection & Preprocessing** – Gather, clean, and transform data into a suitable format.
3. **Model Selection & Training** – Choose an appropriate architecture, train using optimized loss functions, and fine-tune hyperparameters.
4. **Evaluation & Optimization** – Assess performance using relevant metrics and apply techniques like dropout or batch normalization to improve generalization.
5. **Deployment** – Convert and deploy the model via cloud services or APIs for real-world applications.
6. **Monitoring & Maintenance** – Continuously track performance, handle model drift, and update as needed.

2.2.3 The Transformer architecture

The Transformer model is a deep learning architecture introduced by Vaswani et al. in 2017[7]. It is based on the self-attention mechanism, which allows it to process entire sequences in parallel rather than sequentially like recurrent neural networks (RNNs). This architecture has revolutionized natural language processing (NLP) and has been widely adopted in various AI applications.

The key innovation of the Transformer model is the attention mechanism, which enables it to focus on different parts of an input sequence dynamically. Unlike traditional RNNs, Transformers do not rely on sequential processing, making them highly efficient for training on large datasets.

Transformer models have significantly influenced the field of natural language processing (NLP) and have led to major advancements in text understanding, generation, and classification. Below are some widely used Transformer-based architectures:

BERT (Bidirectional Encoder Representations from Transformers)

BERT was introduced by Google in 2018 and is designed to capture bidirectional context in text. Unlike traditional models that read text in a left-to-right or right-to-left manner, BERT understands words based on the surrounding context in both directions. This is achieved through a training method called **Masked Language Modeling (MLM)**, where random words in the text are masked, and the model predicts the missing words based on the surrounding context.

SBERT (Sentence-BERT)

Sentence-BERT (SBERT) is a modification of BERT specifically designed to create semantic embeddings for entire sentences. While BERT processes individual words, SBERT creates vector representations for entire sentences, making it highly effective for tasks that require measuring semantic similarity between sentences.

Key Features:

- Uses Siamese network architecture, allowing it to compare sentence pairs efficiently.
- Provides high-quality sentence embeddings that can be used for various NLP tasks.
- Improves performance in semantic similarity and sentence clustering tasks.

Common Applications:

- Text paraphrase detection (e.g., identifying duplicate questions in forums like Quora).
- Semantic search (e.g., searching for similar articles or documents).
- Chatbots that require understanding conversational intent.

As shown in Figure **??**, SBERT uses a siamese BERT network for classification and similarity scoring.

Unlike traditional BERT, which requires computationally expensive cross-encoders for sentence pair classification, SBERT uses a siamese network with tied weights, allowing it to generate independent sentence representations that can be compared using cosine similarity. The model is fine-tuned using a classification objective, such as on the SNLI dataset, where sentence embeddings are combined using concatenation and element-wise differences before passing through a softmax classifier. At inference, SBERT eliminates the classifier and directly computes similarity scores using cosine similarity, enabling fast retrieval and clustering applications. This architecture significantly reduces computational overhead while maintaining state-of-the-art performance in semantic search, paraphrase detection, and question answering tasks [8].

2.2.4 REST API

A REST API (Representational State Transfer Application Programming Interface) is a web service that follows the principles of REST architecture. It enables communication between client and server applications using standard HTTP methods, making it a widely used approach for building scalable and interoperable web services.

Principles of REST API

A RESTful API follows six key constraints:

- **Statelessness**: Each request from a client to a server must contain all the necessary information, and the server should not store any client session data.
- **Client-Server Architecture**: The client and server are separate entities that interact through requests and responses, ensuring modularity and scalability.
- **Cacheability**: Responses from the server should define whether they are cacheable to optimize performance and reduce network latency.
- **Uniform Interface**: The API should follow consistent standards, such as using standard HTTP methods and resource representations.
- **Layered System**: The architecture can have multiple layers, such as authentication, load balancing, and caching, without affecting client-server interactions.
- **Code on Demand (Optional)**: In some cases, the server can send executable code to the client, such as JavaScript for dynamic functionality.

HTTP Methods in REST API

REST APIs commonly use the following HTTP methods:

- **GET**: Retrieves data from the server.
- **POST**: Sends new data to the server.
- **PUT**: Updates an existing resource on the server.
- **DELETE**: Removes a resource from the server.
- **PATCH**: Partially updates an existing resource.

Response Status Codes

REST APIs return HTTP status codes to indicate the result of a request:

- **200 OK**: The request was successful.
- **201 Created**: A new resource was successfully created.
- **400 Bad Request**: The request was invalid or malformed.

- **401 Unauthorized**: Authentication is required to access the resource.
- **403 Forbidden**: The client does not have permission to access the resource.
- **404 Not Found**: The requested resource does not exist.
- **500 Internal Server Error**: The server encountered an unexpected error.

Common REST API Authentication Methods

- **API Keys**: Clients include an API key in requests for authentication.
- **OAuth 2.0**: A widely used authentication framework for secure authorization.
- **JWT (JSON Web Tokens)**: A compact token-based authentication method.
- **Basic Authentication**: Uses a username and password for authentication (not recommended for production use without HTTPS).

REST API Use Cases

- Web and mobile application backend services.
- Third-party integrations, such as payment gateways and social media APIs.
- IoT (Internet of Things) applications for device communication.
- Cloud services and microservices architecture.

3. Methodology

The development of **Resu-Match** follows a structured workflow integrating Natural Language Processing (NLP), Machine Learning (ML), and Deep Learning (DL) techniques to automate and enhance the recruitment process. The methodology consists of the following key phases:

3.1 Data Collection

The dataset for training and evaluation was sourced from Kaggle, containing resumes categorized into 20 different job categories. Each resume had a corresponding job category label.
Since the dataset lacked job descriptions, we generated relevant job descriptions using the tiiuae/falcon-7b-instruct model via API requests. The generated job descriptions were manually reviewed for relevance and consistency.
To create a high-quality labeled dataset, we manually annotated each resume and job description pair with a matching score ranging from 0 to 1, where:

- **0** represents the lowest relevance, and
- **1** represents a perfect match.

This annotated dataset served as the ground truth for model training and evaluation.

3.2 Data Preprocessing

To ensure the quality and consistency of the data, the following preprocessing steps were applied:

3.2.1 Resume Cleaning

- **Text Extraction**: Resumes were converted from raw formats (PDF, DOCX) into plain text.
- **Noise Removal**: Eliminated unnecessary details such as headers, footers, and extra spaces.
- **Bias Reduction**: Personal details like age, gender, and contact information were anonymized to prevent biases.

3.2.2 Job Description Processing

- **Standardization**: Removed redundant phrases and normalized terminology.
- **Text Cleaning**: Removed irrelevant symbols, HTML tags, and other noise.

3.3 Model Development

The core of the system revolves around sentence similarity matching using a fine-tuned Sentence-BERT (SBERT)[9] model.

3.3.1 Resume and Job Description Embeddings

Instead of traditional keyword-based matching, we leveraged SBERT embeddings to represent resumes and job descriptions as dense vectors in high-dimensional space. This approach enables semantic similarity matching rather than relying on exact keyword matches.
Each resume and job description was transformed into an embedding vector using the custom fine-tuned Sentence-BERT model. The embeddings capture contextual relationships between candidate skills, experience, and job requirements.

3.3.2 Matching and Ranking

The cosine similarity measure was used to compute the degree of match between a resume and a job description. The similarity score determines how well a candidate's profile aligns with the given job requirements.
In addition to embedding-based similarity, we implemented a category-based filtering model that classifies resumes into job categories and ensures that only relevant resumes are compared for ranking.

3.4 Model Evaluation

The system was evaluated using multiple metrics to ensure both accurate and generalizable resume-job description matching.

- **Cosine Similarity Loss**: The model was trained using CosineSimilarityLoss, which optimizes similarity between positive pairs (resume and job description with high relevance) while pushing apart negative pairs. This loss function ensures that embeddings of similar documents are closer in vector space.
- **Spearman's Rank Correlation**: Used for validation, this metric assesses the monotonic relationship between predicted similarity scores and ground truth relevance scores. A higher Spearman correlation suggests that the model correctly ranks resumes in order of relevance.

- **Pearson Correlation Coefficient**: This evaluates the linear relationship between predicted similarity scores and actual relevance scores. It measures how well the model's similarity scores align with human-labeled matching scores.
- **F1-score**: This metric balances precision and recall, ensuring that the model correctly identifies relevant resumes while minimizing false positives and negatives. A higher F1-score indicates a well-performing model that effectively distinguishes between relevant and non-relevant resumes.

To validate the model's generalization, it was tested on unseen resume-job description pairs from the dataset. The combination of these metrics provides a comprehensive assessment of both ranking accuracy and overall similarity predictions.

Epoch	Step	Validation Pearson Cosine	Validation Spearman Cosine
1	100	0.855952	0.835985
	108	0.864010	0.851528
2	100	0.844784	0.835854
	108	0.838382	0.826938
3	100	0.862289	0.864927
	108	0.869847	0.865006
4	100	0.866819	0.878287
	108	0.857751	0.862626
5	100	0.868232	0.857564
	108	0.874660	0.867557
6	100	0.875579	0.872081
	108	0.881708	0.874093
7	100	0.855295	0.858247
	108	0.848166	0.847254
8	100	0.862272	0.847938
	108	0.874871	0.860338

Table 3.1: Validation Cosine Similarity Metrics for Each Epoch and Step

3.5 Model Optimization

To improve both performance and efficiency, we employed multiple optimization techniques. These optimizations aimed to enhance the model's accuracy, reduce computation time, and ensure scalability for real-time resume-job description matching.

3.5.1 Hyperparameter Tuning

Hyperparameter tuning plays a crucial role in optimizing model performance. We experimented with various hyperparameters to find the best configuration for our task:

- **Batch Size**: Adjusting batch size impacts training stability and memory usage.
  - Smaller batch sizes (e.g., 8 or 16) lead to more frequent updates but can introduce higher variance in gradients, making training less stable.
  - Larger batch sizes (e.g., 32 or 64) provide more stable gradient updates but require more memory and may lead to slower convergence.
- **Learning Rate**: Determines the rate at which model weights are updated during training.
  - A high learning rate (e.g., 5e-4) can cause the model to converge quickly but may lead to instability.
  - A very low learning rate (e.g., 1e-6) ensures stable updates but may slow down training significantly.
  - We applied learning rate scheduling, starting with a higher learning rate and gradually reducing it for effective fine-tuning.
- **Embedding Dimensions**: Defines the vector size of text representations.
  - Higher-dimensional embeddings (e.g., 768 or 1024) capture more features but increase computational cost.
  - Lower-dimensional embeddings (e.g., 256 or 512) reduce memory usage but may lose important contextual information.

3.5.2 Fine-Tuning on Custom Data

The Sentence-BERT model was fine-tuned on a custom annotated dataset to adapt it for resume-job description matching.

- Fine-tuning helps the model learn domain-specific terminology and ranking patterns used by recruiters.
- The model was trained on resume-job description pairs labeled with a matching score (0-1).
- The training objective was to maximize similarity for relevant matches while minimizing it for non-relevant ones using CosineSimilarityLoss.

3.5.3 Efficient Inference

To ensure real-time resume processing, we employed the following optimizations:

- **Sentence-BERT Instead of BERT**:
  - Traditional BERT processes text pairs together, making large-scale matching expensive.
  - Sentence-BERT enables independent encoding of resumes and job descriptions, significantly reducing computation time.
- **Vectorized Similarity Computation**:
  - Used batch processing and cosine similarity calculations to process multiple resumes at once.
- **Hardware Optimization**:
  - Leveraged GPU acceleration via PyTorch's CUDA for faster inference.
  - Optimized CPU-based execution using Faiss and `NumPy` for efficient similarity computations.

The combination of these optimizations resulted in improved accuracy, faster inference, and scalable real-time resume processing.

3.6 System Integration

The system was deployed as a web application tailored for recruiters, offering an intuitive interface for uploading resumes, evaluating matching scores, and exporting results. The architecture consists of a FastAPI backend for efficient data processing and a React-based frontend for an interactive user experience.

3.6.1 Backend

The backend was developed using FastAPI, a high-performance framework for building scalable APIs. Key features include:

- **Lightweight and Fast**: FastAPI ensures low-latency processing, suitable for real-time resume-job matching.
- **Asynchronous Processing**: Supports concurrent requests using async and await, improving response times.

- **Seamless Model Integration**: Connects efficiently with the Sentence-BERT model for embedding and similarity computations.

Key API Endpoints:

- `/upload_resumes` – Accepts multiple resumes for processing.
- `/match_resumes` – Computes similarity scores based on Sentence-BERT embeddings.
- `/download_results` – Generates downloadable reports in Excel format.

3.6.2 Frontend

The frontend was built using React, providing a fast and interactive experience with:

- **Component-Based Architecture**: Ensures modularity and easy feature expansion.
- **Fast Rendering**: React's Virtual DOM optimizes performance for large datasets.
- **Seamless API Integration**: Communicates with FastAPI to fetch and display results dynamically.

3.6.3 Recruiter Dashboard Features

- **Upload Resumes & Job Description**: Recruiters can upload multiple resumes (PDF) and a single job description.
- **Matching Score Display**: Similarity scores are calculated and ranked for each resume.
- **Download Options**: Allows downloading individual resumes or exporting results in Excel format.

3.6.4 Future Enhancements (Production Version)

- **User Mode for Job Seekers**: Candidates will be able to upload resumes and receive job recommendations.
- **Explainability Features**: Future versions may highlight keywords and provide visualization tools for score interpretation.

3.7 Performance and Scalability

The system is designed to handle real-time resume processing, making it suitable for medium-to-large-scale recruitment. It can efficiently match and rank up to 1000 resumes in 5-7 minutes, ensuring quick turnaround times for recruiters handling bulk applications. The architecture is optimized for both speed and scalability, allowing seamless deployment in enterprise hiring workflows.

3.7.1 Real-Time Processing Capability

The system achieves real-time processing through the following optimizations:

- **Optimized Sentence-BERT embeddings**: The use of Sentence-BERT enables fast computation of semantic similarity, reducing processing time compared to traditional BERT models.
- **Batch Processing**: Resumes are processed in parallel batches, significantly improving computational efficiency.
- **Asynchronous API Calls**: The FastAPI backend supports asynchronous execution, allowing multiple requests to be processed simultaneously without blocking system operations.

3.7.2 Scalability Considerations

To ensure the system can handle increasing workloads, several scalability measures have been implemented:

- **Horizontal Scaling**: Multiple backend instances can be deployed to process resumes in parallel, ensuring smooth performance under high loads.
- **Efficient Memory Management**: The system optimizes memory usage by processing only the necessary embeddings at a time, minimizing resource consumption.
- **Cloud Deployment Potential**: The architecture supports deployment on cloud platforms (AWS, Azure, or GCP) with auto-scaling capabilities to dynamically allocate resources based on demand.

3.7.3 Bias Mitigation in Resume Screening

The use of machine learning in many industries has raised many ethical and legal concerns, especially that of fairness and bias in predictions.([10] [11]) Traditional hiring processes often involve human screening of resumes, which may introduce unconscious bias based on:

- Name, gender, or ethnicity inferred from personal details.
- Educational institutions or location influencing recruiter decisions.
- Resume formatting and stylistic differences affecting subjective judgment.

By leveraging an AI-powered approach, our system ensures:

- **Objective Evaluation**: The model assesses resumes solely based on skill and experience relevance, removing human biases.
- **Reduced Manual Effort**: Recruiters can focus on high-scoring candidates without manually screening every resume.
- **Fairer Hiring Decisions**: Each resume is processed uniformly, ensuring consistency and impartiality in evaluation.

3.7.4 Enhancing the Recruitment Process

By combining speed, scalability, and unbiased evaluation, the system significantly improves the recruitment workflow:

- **Faster Hiring Decisions**: Reduces time spent on manual screening, allowing quicker shortlisting.
- **Increased Efficiency**: Enables recruiters to focus on high-quality candidates rather than filtering hundreds of applications.
- **Scalability for Large Organizations**: The system is adaptable for both small firms and large enterprises handling thousands of applications.

4. System design

4.1 System Architecture

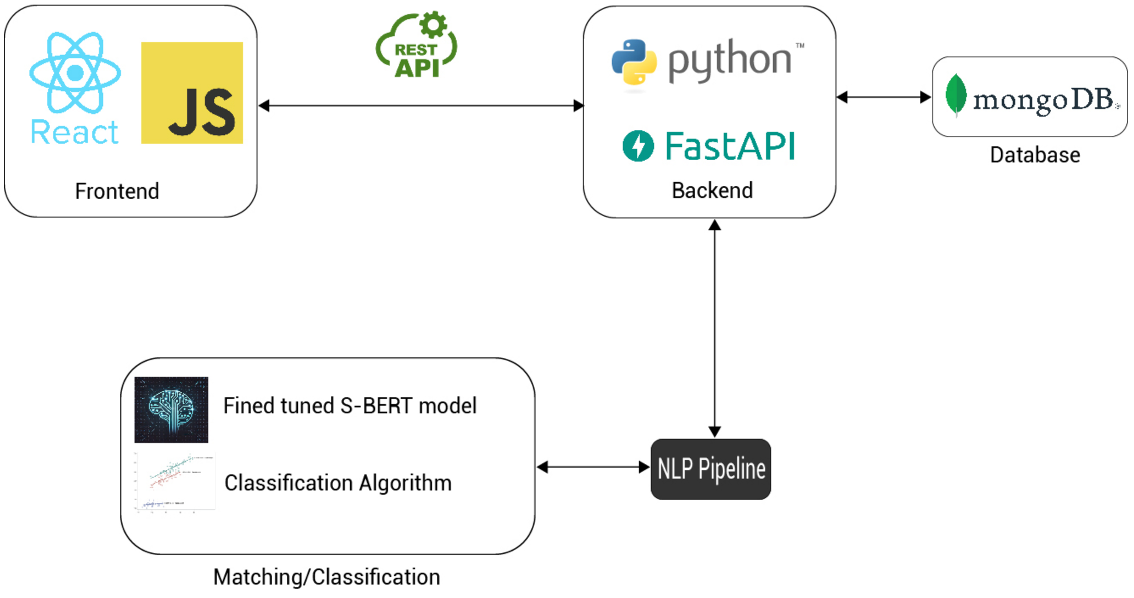


Figure 4.1: System Architecture

4.2 System Diagrams

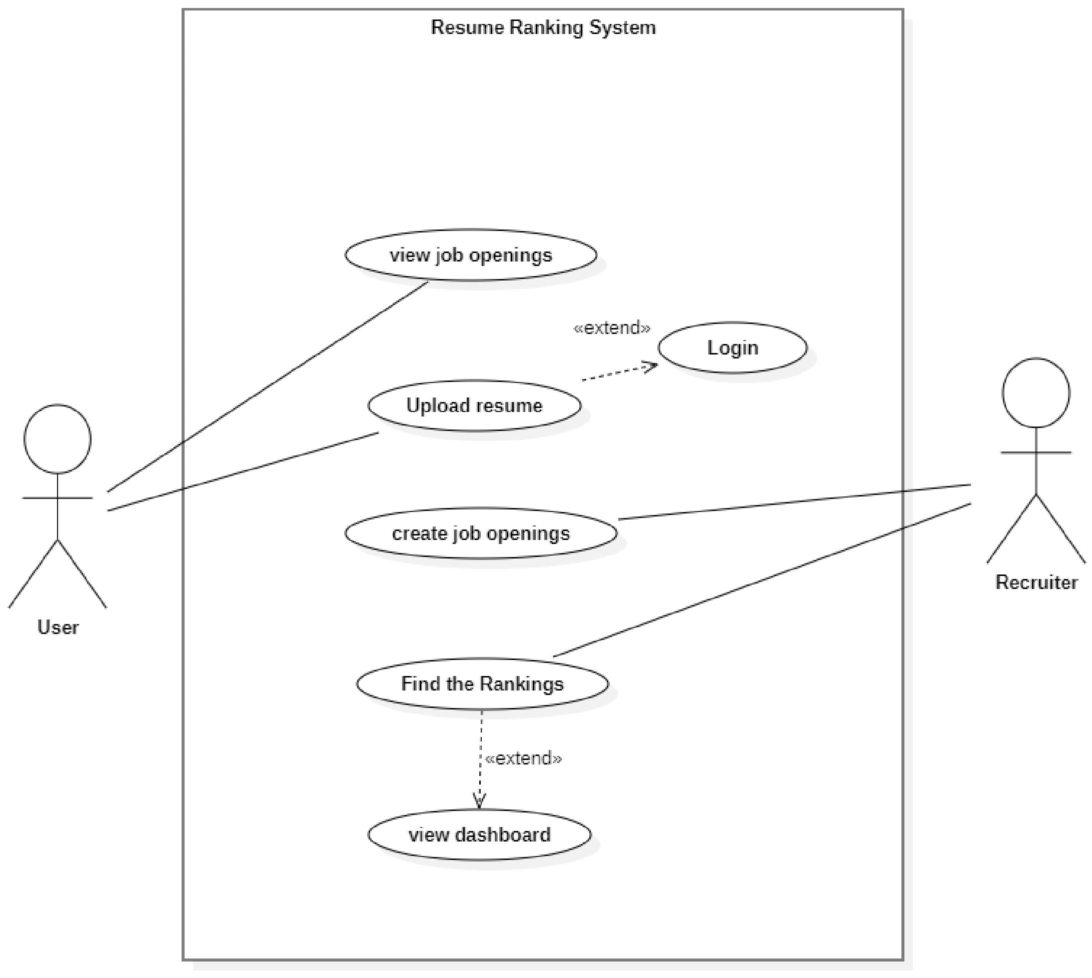


Figure 4.2: Use Case Diagram

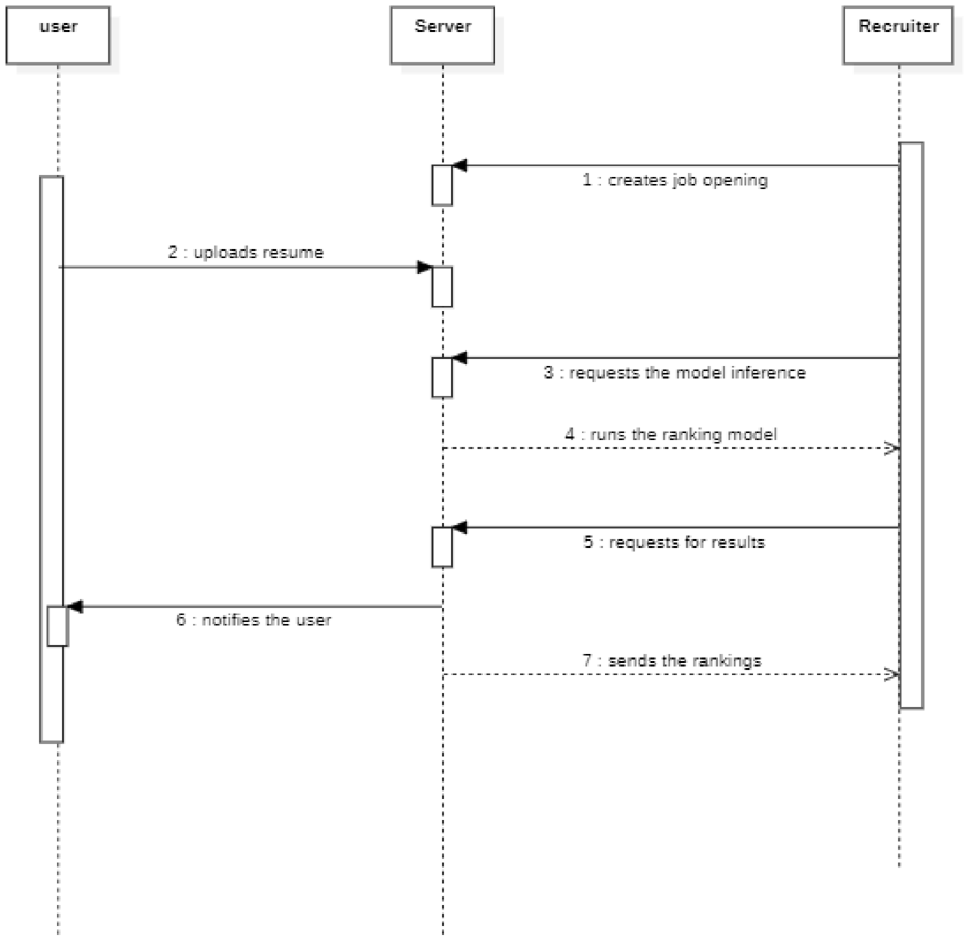


Figure 4.3: Sequence Diagram

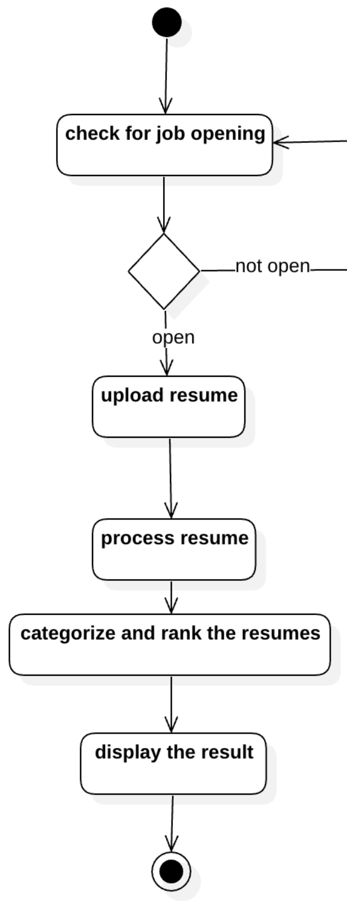


Figure 4.4: Activity Diagram

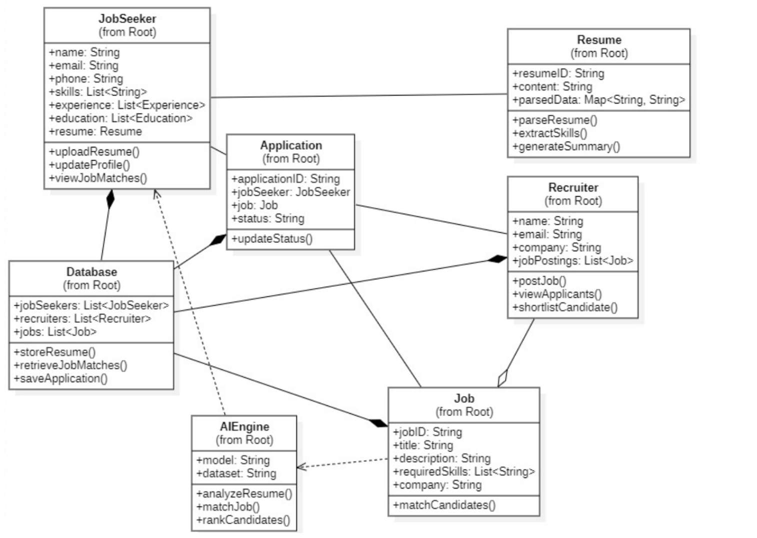


Figure 4.5: Class Diagram

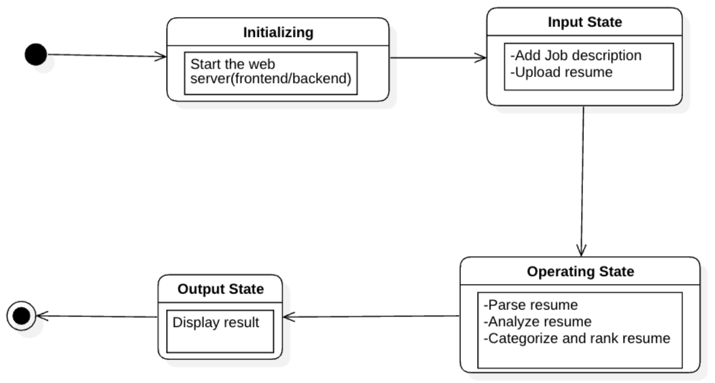


Figure 4.6: State Diagram

5. Results & Discussion

5.1 Results & Discussion

5.1.1 Results

The implementation of Resu-Match successfully automated the resume screening and ranking process using AI. The system utilized a fine-tuned SBERT model to generate embeddings and compute similarity scores between resumes and job descriptions. The results obtained from multiple validation metrics demonstrated the effectiveness of the model in matching candidates to job roles.
Key results include:

- **High Accuracy in Resume Matching**: The model achieved a validation cosine similarity score of up to 0.88 and Pearson correlation coefficient of 0.87, indicating strong agreement between predicted and actual relevance scores.

- **Efficient Processing Time**: The system processed up to 1000 resumes in under 7 minutes, demonstrating its scalability for bulk recruitment scenarios.

- **Bias Mitigation**: The removal of personal information during preprocessing helped ensure fair candidate evaluation.

- **User-Friendly Experience**: The recruiter dashboard provided a seamless interface for uploading job descriptions, reviewing ranked candidates, and exporting results.

- **Regression and Classification Performance**:
  - Mean Squared Error (MSE): **0.0230**
  - Mean Absolute Error (MAE): **0.1235**
  - Accuracy: **0.8708**
  - Precision: **0.9733**
  - Recall: **0.8922**
  - F1 Score: **0.9309**

5.1.2 Discussion

The results indicate that Resu-Match provides a robust and efficient solution for AI-powered recruitment. The combination of SBERT-based semantic similarity ranking and machine learning classification proved effective in reducing recruitment time and improving candidate selection accuracy.

Model Performance Analysis

- **Semantic Understanding**: The SBERT model effectively captured contextual relationships between resumes and job descriptions, unlike traditional keyword-based matching systems.
- **Validation Trends**: The Spearman's rank correlation consistently remained above 0.85, confirming that the system correctly ranked resumes in order of relevance.
- **Regression Metrics**: The low MSE (0.0230) and MAE (0.1235) indicate that the model provides highly accurate similarity score predictions.
- **Classification Metrics**: The model achieves an impressive F1-score of 0.9309, demonstrating a balance between precision and recall.
- **Generalization Ability**: The model performed well on unseen resume-job description pairs, indicating strong generalization to real-world data.

Strengths and Contributions

- **Automation of Resume Screening**: Eliminated the need for manual shortlisting, allowing recruiters to focus on top candidates.
- **Scalability for Large Enterprises**: The system efficiently handled bulk resume processing, making it viable for large-scale hiring.
- **Reduction of Hiring Bias**: By anonymizing personal details, the system promoted fair and unbiased candidate evaluations.

Limitations

- **Complex Resumes**: The model occasionally struggled with unconventional resume formats, requiring further preprocessing improvements.

- **Industry-Specific Matching**: While effective in general recruitment, fine-tuning for industry-specific hiring criteria could enhance precision.
- **Transparency of Ranking**: Recruiters may require additional insights into how similarity scores are computed.

5.1.3 Evaluation Metrics

The model was assessed using:

1. **Cosine Similarity Score**: Evaluating how closely resumes matched job descriptions.
2. **Spearman's Rank Correlation**: Measuring the monotonic relationship between predicted and actual rankings.
3. **Pearson Correlation Coefficient**: Determining the linear relationship between predicted and ground truth scores.
4. **Mean Squared Error (MSE)**: Quantifying the average squared differences between predicted and actual similarity scores.
5. **Mean Absolute Error (MAE)**: Measuring the average absolute differences between predictions and actual values.
6. **Precision-Recall Metrics**: Evaluating model performance in classifying relevant resumes.

Metric	**Best Score Achieved**
Validation Cosine Similarity	**0.88**
Pearson Correlation	**0.87**
Spearman's Rank Correlation	**0.85+**
Mean Squared Error (MSE)	**0.0230**
Mean Absolute Error (MAE)	**0.1235**
Accuracy	**0.8708**
Precision	**0.9733**
Recall	**0.8922**
F1 Score	**0.9309**

Table 5.1: Performance Metrics for Resu-Match System

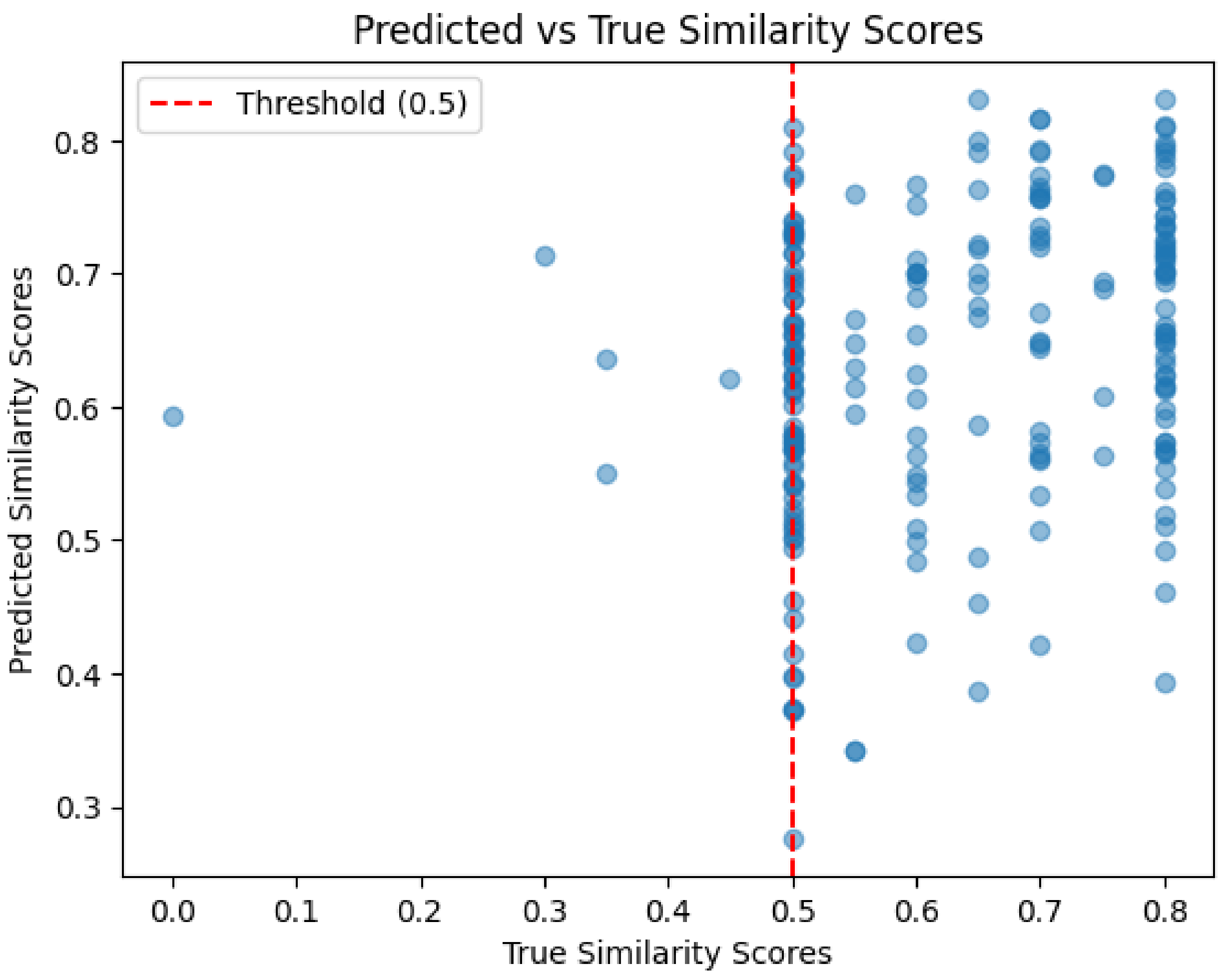


Figure 5.1: Evaluation with 0.5 threshold

Validation Strategy

- **Cross-Validation**: The dataset was split into 80% training and 20% validation sets to ensure model robustness.
- **Real-World Testing**: The system was tested using actual job descriptions from recruiters to validate performance.
- **Bias and Fairness Checks**: Ensured that anonymized resumes were ranked based on skills and experience rather than personal attributes.

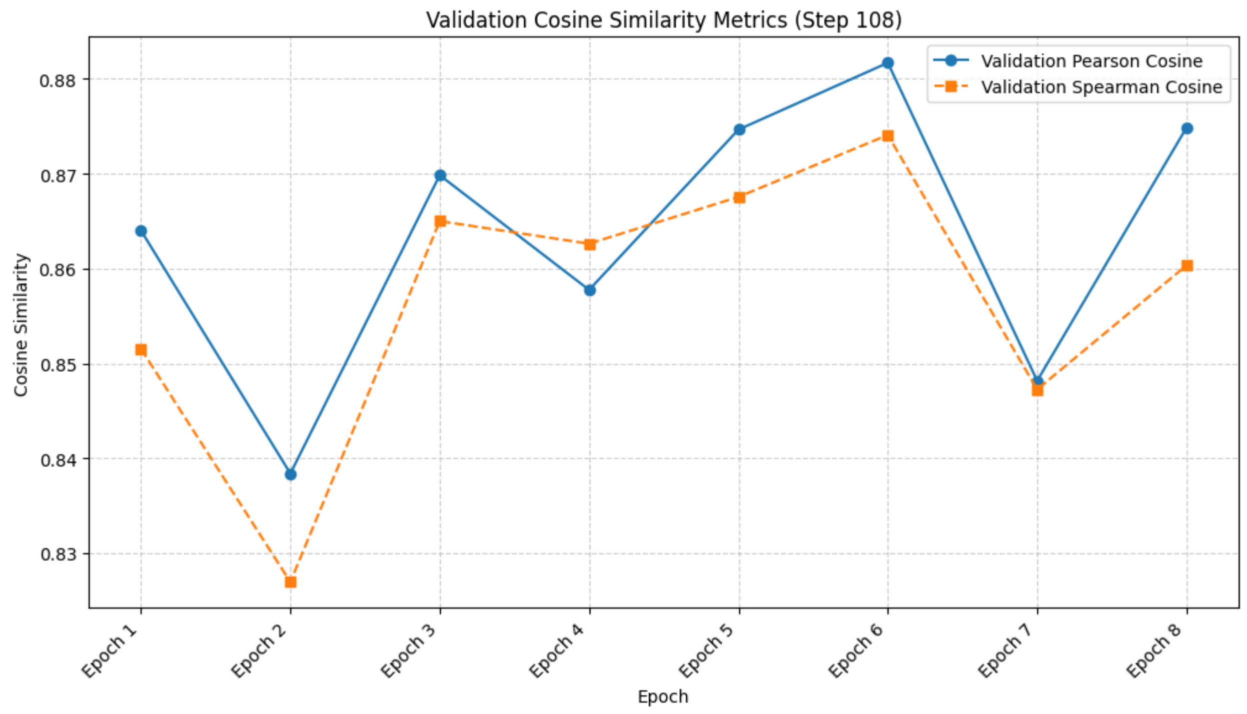


Figure 5.2: Validation cosine similarity metrics

5.1.4 Testing on resume of Batchmates

As instructed by the project supervisor the model was tested on the actual resumes of our batch-mates. Unfortunately, we could onlygather the resume of very few of them. The results achieved from that test is shown in the diagrams below.

- **Accuracy**: 0.62 (62%)
  - The model correctly classifies 62% of all instances
- **Precision**: 0.58 (58%)
  - When the model predicts a class, it's correct 58% of the time
- **Recall**: 0.62 (62%)
  - The model identifies 62% of all actual positive cases
- **F1-Score**: 0.59
  - Harmonic mean of precision and recall

The results of the testing on the resumes of our batch-mates could not yield optimum results as the model was trained on around 20 job categories from different domain. As many of

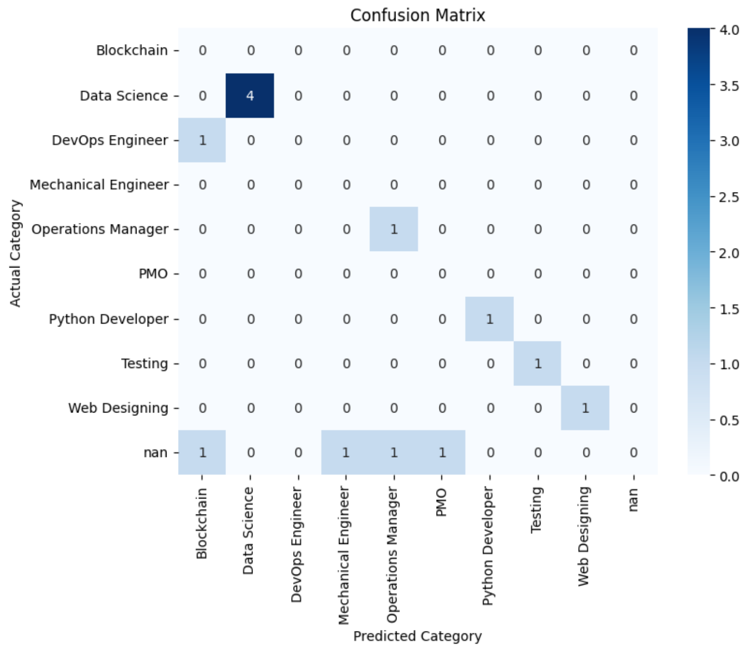


Figure 5.3: Confusion Matrix

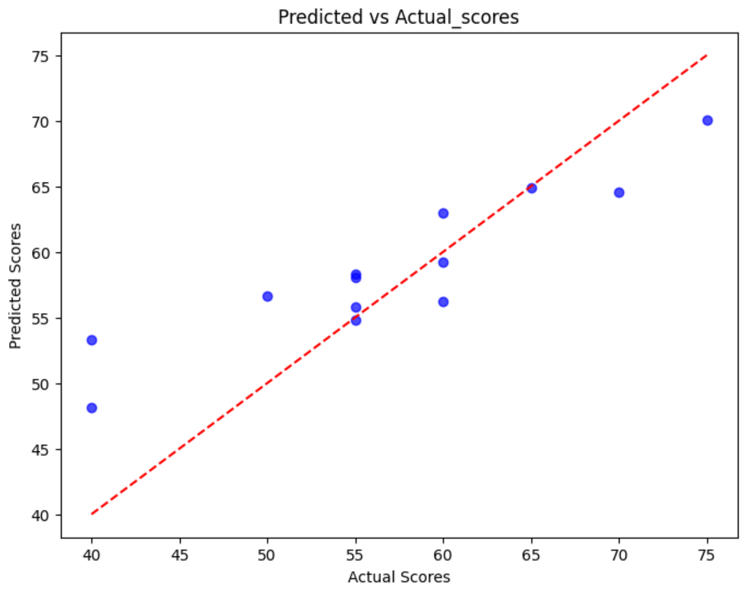


Figure 5.4: Predicted vs actual score

them had their resumes inclined to different fields which were out of the training horizon of the model's 20 categories. Though the results in the score prediction side were quite good as here more than the numeric accuracy the generalization and ranking of the resume was of utmost priority.

6. Conclusions

The **Resu-Match AI Recruitment System** has been successfully developed to enhance and streamline the resume screening process through the integration of artificial intelligence (AI), natural language processing (NLP), and machine learning (ML) algorithms. The system effectively automates candidate evaluation, reducing manual effort while improving efficiency, accuracy, and fairness in recruitment. It enables automated resume parsing and analysis to extract relevant candidate information efficiently, while intelligent skill-matching algorithms align candidates with job requirements based on experience, qualifications, and competencies. Additionally, bias mitigation techniques have been implemented to ensure fair and objective assessments, and the integration of data analytics dashboards provides recruiters with actionable insights, facilitating data-driven decision-making.

While the system demonstrates significant improvements in the recruitment process, certain challenges were identified, including handling diverse resume formats, improving contextual understanding, and enhancing the explainability of AI-driven decisions. Addressing these challenges in future iterations will require incorporating advanced AI models, real-time learning capabilities, and enhanced user feedback mechanisms to further refine the system's accuracy and usability.

In conclusion, the **Resu-Match AI Recruitment System** presents a scalable and efficient solution to modern hiring challenges. By leveraging emerging AI technologies and continuously refining its features, the system has the potential to revolutionize recruitment processes, making them more objective, time-efficient, and data-driven. With continued advancements, Resu-Match can serve as a transformative tool in the hiring landscape, ensuring more effective talent acquisition and improved hiring outcomes.

7. Limitations and Future enhancement

7.1 Limitations

The Resu-Match system enhances automated resume screening but has limitations affecting accuracy, efficiency, and deployment. These arise from data availability, model performance, computational demands, ethical concerns, and integration challenges, requiring careful consideration for improvement.

- **Data Limitations:** The system's effectiveness depends on the dataset's quality and diversity. The Kaggle dataset may not capture real-world hiring complexities, limiting generalization to niche roles. AI-generated job descriptions may not fully reflect the specificity of actual postings.

- **Model Performance Constraints:** Despite using SBERT embeddings, the model may misinterpret subtle differences in job requirements and resumes. The model relies on industry-specific keywords, potentially undervaluing qualified candidates with non-standard terminology. Overfitting to the training dataset further limits its ability to generalize to new data.

- **Computational and Scalability Challenges:** Processing large resume datasets requires high computational power. While optimized, handling over 100,000 resumes can slow performance, necessitating GPU acceleration. Real-time processing may experience latency, affecting recruiter's decision-making.

- **Ethical and Fairness Concerns:** Although personal details are anonymized, biases from historical hiring data may persist. The model lacks interpretability, making it difficult to justify ranking decisions. Job seekers might manipulate resumes with keywords to improve scores without possessing relevant skills.

- **Practical Deployment Limitations:** Integrating with existing Applicant Tracking Systems (ATS) may require extensive customization. Recruiters may resist AI-driven screening due to accuracy concerns. Data privacy is also critical, requiring compliance with regulations like GDPR to protect candidate information.

7.2 Future Enhancements

To improve Resu-Match, several enhancements are planned to enhance accuracy, efficiency, and usability. Future versions will incorporate advanced transformer models like GPT-based architectures to improve contextual understanding. The system will be trained on a larger and more diverse dataset covering additional job categories and industries, incorporating real-world job descriptions to better reflect hiring trends. Real-time job recommendations will be introduced, enabling candidates to receive relevant job suggestions instantly while recruiters can identify top-matching applicants efficiently. Bias mitigation strategies, such as adversarial debiasing and explainable AI, will be implemented to ensure fairness and transparency in candidate evaluations. Additionally, Resu-Match will be designed for seamless integration with Applicant Tracking Systems (ATS) through standardized APIs, while optimized cloud deployment options will enhance scalability and accessibility for enterprise users. These enhancements will make the system more robust, fair, and effective for modern recruitment needs.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT*, 2018.

[2] Tom B. Brown et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[3] Li et al. Job description parsing with deep learning. *Journal of AI Research*, 2023.

[4] Smith et al. Improving recruitment processes with machine learning models. *International Journal of Computer Science*, 2022.

[5] García and González. Ai and recruitment: Benefits and ethical challenges. *International Journal of Artificial Intelligence in Education*, 2022.

[6] Rohit Bakoliya. sieve.ai: Ai-powered resume screening tool, 2025. Accessed: 2025-03-02.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[8] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[9] Nils Reimers and Iryna Gurevych. Sentence-transformers: Multilingual sentence embeddings using bert xlnet, 2025. Accessed: 2025-03-02.

[10] Andrew D. Selbst, Danah Boyd, Sorelle A. Friedler, Suresh Venkatasubramanian, and Janet Vertesi. Fairness and abstraction in sociotechnical systems. *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT)*, page 59–68, 2019.

[11] Solon Barocas, Moritz Hardt, and Arvind Narayanan. Fairness and machine learning. *arXiv preprint arXiv:1908.09635*, 2019.

Appendix

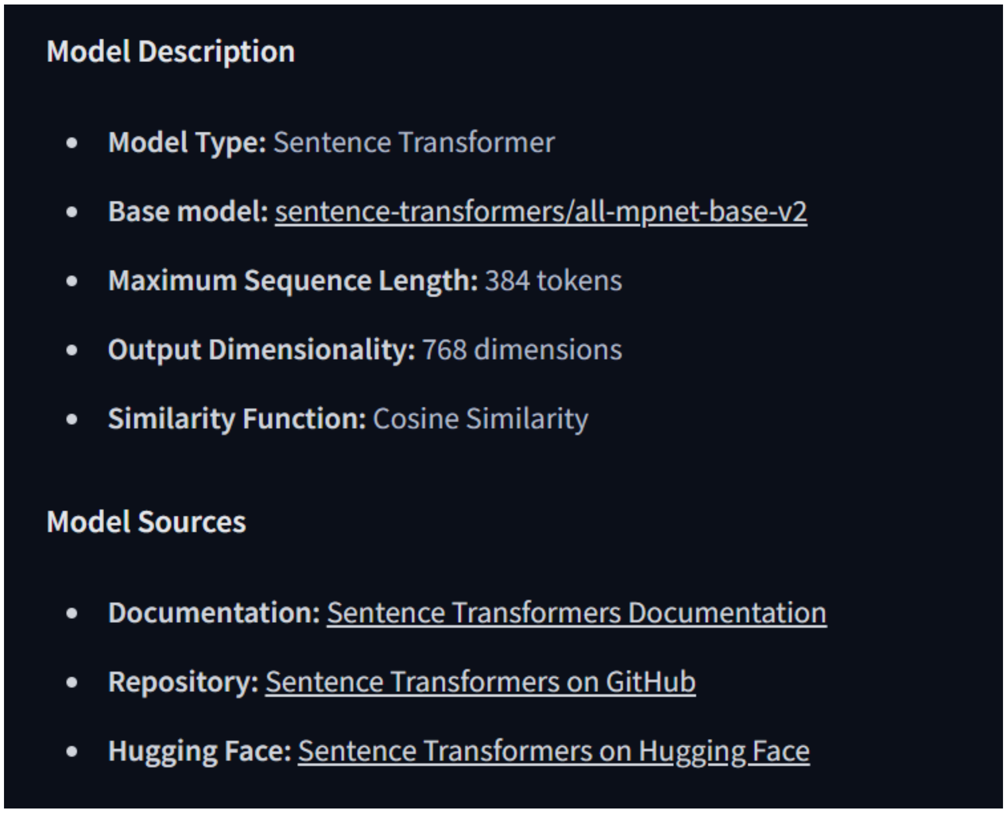
Source Code

The source code for this project can be found at https://github.com/AshimSapkota/minor_project

Model

The fine-tuned model can be found at https://huggingface.co/maashimho/tuned_for_project

Model description



Model Description

- **Model Type:** Sentence Transformer
- **Base model:** sentence-transformers/all-mpnet-base-v2
- **Maximum Sequence Length:** 384 tokens
- **Output Dimensionality:** 768 dimensions
- **Similarity Function:** Cosine Similarity

Model Sources

- **Documentation:** Sentence Transformers Documentation
- **Repository:** Sentence Transformers on GitHub
- **Hugging Face:** Sentence Transformers on Hugging Face

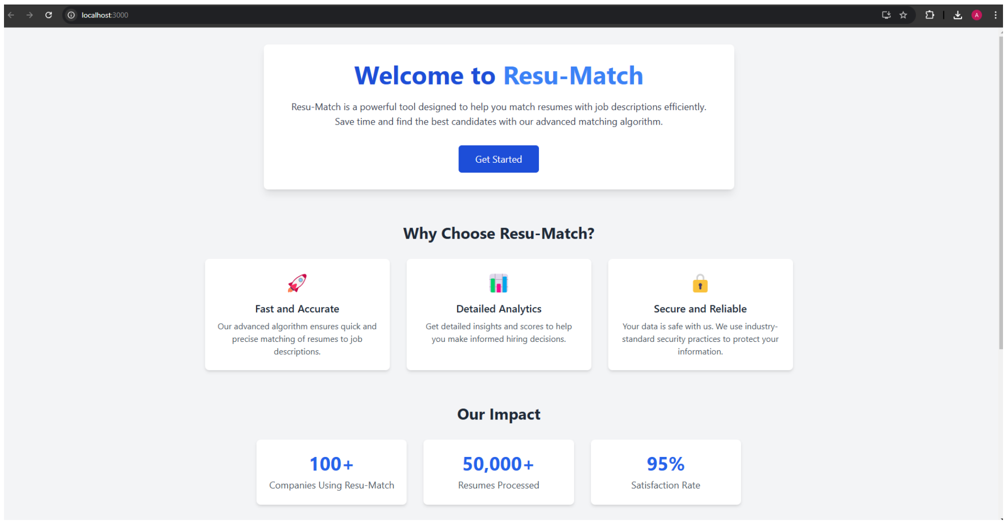
Figure 7.1: Model Description

Can be used by:

```python
from sentence_transformers import SentenceTransformer

model = SentenceTransformer("maashimho/tuned_for_project")
```

Webapp


localhost:3000

Welcome to Resu-Match

Resu-Match is a powerful tool designed to help you match resumes with job descriptions efficiently. Save time and find the best candidates with our advanced matching algorithm.

Why Choose Resu-Match?

Fast and Accurate

Our advanced algorithm ensures quick and precise matching of resumes to job descriptions.

Detailed Analytics

Get detailed insights and scores to help you make informed hiring decisions.

Secure and Reliable

Your data is safe with us. We use industry-standard security practices to protect your information.

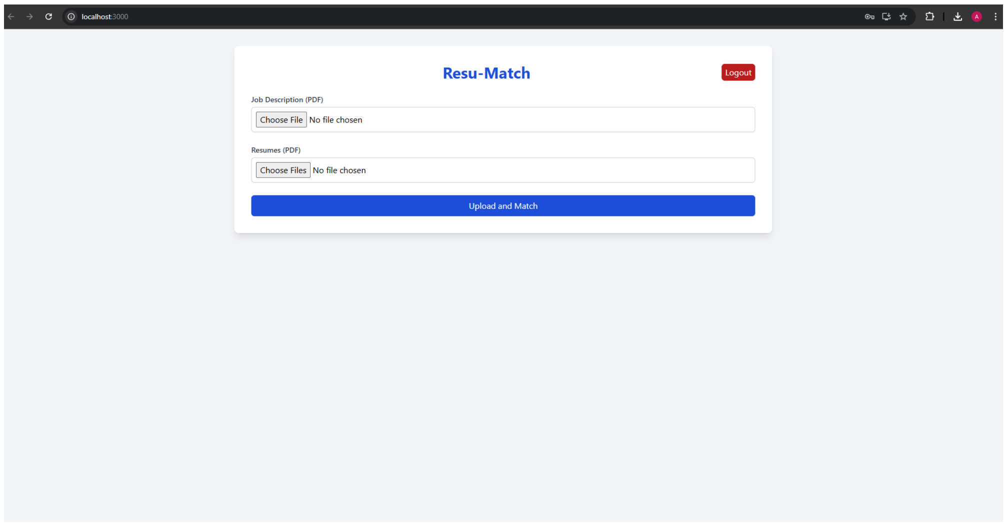
Our Impact

100+ Companies Using Resu-Match

50,000+ Resumes Processed

95% Satisfaction Rate

Figure 7.2: Home Page


localhost:3000

Resu-Match

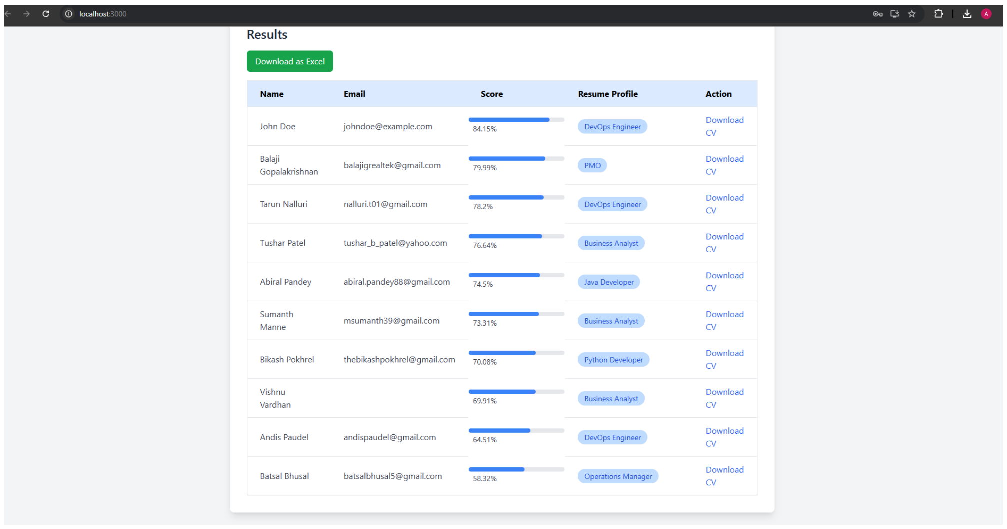
Job Description (PDF)

No file chosen

Resumes (PDF)

No file chosen

Figure 7.3: Admin Interface



localhost:3000

Results

Download as Excel

Name	Email	Score	Resume Profile	Action
John Doe	johndoe@example.com	84.15%	DevOps Engineer	Download CV
Balaji Gopalakrishnan	balajigrealtek@gmail.com	79.99%	PMO	Download CV
Tarun Nalluri	nalluri.t01@gmail.com	78.2%	DevOps Engineer	Download CV
Tushar Patel	tushar_b_patel@yahoo.com	76.64%	Business Analyst	Download CV
Abiral Pandey	abiral.pandey88@gmail.com	74.5%	Java Developer	Download CV
Sumanth Manne	msumanth39@gmail.com	73.31%	Business Analyst	Download CV
Bikash Pokhrel	thebikashpokhrel@gmail.com	70.08%	Python Developer	Download CV
Vishnu Vardhan		69.91%	Business Analyst	Download CV
Andis Paudel	andispaudel@gmail.com	64.51%	DevOps Engineer	Download CV
Batsal Bhusal	batsalbhusal5@gmail.com	58.32%	Operations Manager	Download CV

Figure 7.4: Admin Dashboard