

AI-powered Resume Screening and Ranking System

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning

with

TechSaksham – A joint CSR initiative of Microsoft & SAP

by

Name - Rajesh Pal,

Email – rajeshpal31082001@gmail.com

Under the Guidance of

Saomya Chaudhury

ACKNOWLEDGEMENT

I would like to take this opportunity to express my heartfelt gratitude to everyone who has supported me throughout this project.

First and foremost, I extend my sincere thanks to my supervisor, **Saomya Chaudhury**, for his invaluable guidance, encouragement, and continuous support. His insightful advice, constructive feedback, and unwavering confidence in me have been instrumental in shaping this project. His mentorship has not only helped me successfully complete this work but has also inspired me to think innovatively and improve as a professional.

Working under his guidance for the past year has been an enriching experience. His support extended beyond just the technical aspects of the project, helping me grow in multiple dimensions. I am truly grateful for his patience, motivation, and the knowledge he has shared with me.

ABSTRACT

In today's job market, recruiters often receive a large number of resumes for a single job posting, making manual screening a tedious and time-consuming task. To streamline this process, our project introduces an **AI-based Resume Ranking System** that automatically evaluates and ranks resumes based on their relevance to a given job description.

The goal of this project is to develop an efficient and accurate system that helps recruiters quickly identify the most suitable candidates. The system applies **Natural Language Processing (NLP)** and **Machine Learning (ML)** techniques to analyze resumes and match them against job descriptions.

The process begins with **document parsing**, followed by **text preprocessing**, where unnecessary elements like stopwords and special characters are removed. Next, **feature extraction** is performed using the **TF-IDF (Term Frequency-Inverse Document Frequency)** method. The extracted features are then compared using **cosine similarity**, which measures how closely a resume aligns with the job description. Finally, the system ranks the top **10 most relevant resumes** using the **K-Nearest Neighbors (KNN) algorithm**.

The results demonstrate that the system improves the accuracy and efficiency of the hiring process by reducing the time spent on manual resume screening. By automating this step, recruiters can make faster, unbiased, and more informed decisions.

In conclusion, this project presents a **practical and scalable** solution for resume shortlisting. Future improvements may include **integrating deep learning models**, handling **multiple job descriptions simultaneously**, and enhancing ranking accuracy with more advanced **semantic analysis techniques**.

TABLE OF CONTENT

Abstract	I
Chapter 1. Introduction	1
1.1 Problem Statement	1
1.2 Motivation	2
1.3 Objectives	3
1.4 Scope of the Project	4
Chapter 2. Literature Survey	5
2.1 Review of Relevant Literature	5
2.2 Existing Models, Techniques, and Methodologies	7
2.3 Gaps in Existing Solutions and How This Project Addresses	10
Chapter 3. Proposed Methodology	12
3.1 System Design Explanation	12
3.2 Requirement Specification	14
Chapter 4. Implementation and Results	16
4.1 Snapshot of the Project	16
4.2 Github Link for the Project	18
Chapter 5. Discussion and Conclusion	19
5.1 Future Work	19
5.2 Conclusion	21
References	22

LIST OF FIGURES

Figure No.	Figure Caption	Page No.
Figure 1	System Architecture	12
Figure 2	Input Snapshot	16
Figure 3	Output Snapshot	17

LIST OF TABLES

Table. No.	Table Caption	Page No.
Table 1	Comparison of Existing Models	7

CHAPTER 1

Introduction

1.1 Problem Statement

The traditional hiring process involves manually reviewing a large number of resumes to identify candidates who best fit a given job description. This approach is not only time-consuming but also prone to human biases and inconsistencies, leading to inefficient recruitment decisions. Recruiters often struggle to shortlist candidates effectively due to the overwhelming volume of applications, which may result in overlooking highly qualified individuals.

Additionally, different job roles require specific skills, qualifications, and experience, making it difficult to manually compare and rank resumes accurately. The lack of a standardized method for evaluating resumes against job descriptions further complicates the process. As a result, companies face delays in hiring the right talent, which can negatively impact productivity and business growth.

To address these challenges, there is a need for an automated resume ranking system that can analyze resumes based on job-specific criteria, providing recruiters with an efficient, objective, and scalable solution for shortlisting candidates. This system should leverage natural language processing (NLP) and machine learning techniques to compare resumes with job descriptions, ensuring a fair and streamlined recruitment process.

1.2 Motivation

The primary motivation behind this project is to enhance the efficiency and accuracy of the recruitment process by leveraging automation. Traditional resume screening methods are time-consuming and often subjective, leading to inefficiencies in candidate selection. With the increasing number of job applicants for every position, recruiters need an intelligent system that can assist in shortlisting the most suitable candidates quickly and fairly.

This project was chosen to address these challenges by developing an AI-powered resume ranking system that can analyze resumes based on job descriptions, ensuring that the best-fit candidates are identified without manual intervention. By integrating Natural Language Processing (NLP) and machine learning techniques, the system can assess resumes based on relevant skills, experience, and qualifications, reducing the risk of human bias in the hiring process.

1.2.1 Potential Applications and Impact

- **Faster Recruitment Process:** Automates resume screening, reducing the time spent on manual shortlisting.
- **Improved Hiring Accuracy:** Uses AI to rank candidates based on their suitability for the job, ensuring better hiring decisions.
- **Objective Candidate Evaluation:** Minimizes human bias by evaluating resumes purely on skill matching and relevance to job descriptions.
- **Scalability for Large Applications:** Can be applied in companies with high job applications, reducing workload on recruiters.
- **Enhanced Candidate Experience:** Speeds up response time for job applicants by streamlining the screening process.
- **Potential Integration with HR Systems:** Can be integrated into applicant tracking systems (ATS) to further optimize hiring workflows.

1.3 Objective

The primary objective of this project is to develop an AI-driven system that ranks resumes based on job descriptions, ensuring a more efficient and unbiased recruitment process. The specific objectives include:

- **Automated Resume Screening:** Design a system that automatically evaluates resumes based on job requirements.
- **Efficient Candidate Ranking:** Implement a ranking mechanism that prioritizes resumes based on their relevance to the job description.
- **Reduction of Manual Effort:** Minimize the time and effort required by recruiters in the initial screening phase.
- **Unbiased Selection Process:** Ensure fair evaluation by relying on AI-driven analysis rather than human judgment.
- **Integration with NLP and Machine Learning:** Utilize NLP techniques to extract key skills, experience, and qualifications from resumes.
- **User-Friendly Interface:** Provide an easy-to-use platform where recruiters can upload job descriptions and resumes for quick analysis.
- **Scalability and Flexibility:** Ensure the system can handle large volumes of resumes and adapt to different job roles.

1.4 Scope of the Project

1.4.1 Scope:

This project aims to develop an AI-based resume ranking system that enhances the recruitment process by automatically evaluating and ranking resumes based on job descriptions. The system leverages **Natural Language Processing (NLP)** and **Machine Learning (ML)** to analyze resume content and match it with the required qualifications and skills for a job.

Key aspects covered in the project include:

- **Automated Resume Parsing:** Extract relevant information such as skills, experience, and education from resumes.
- **Job Description Analysis:** Process job descriptions to identify key requirements.
- **Resume-Job Matching:** Compare resumes with job descriptions using NLP techniques.
- **Ranking Mechanism:** Assign scores to resumes based on relevance and suitability for the job.
- **User-Friendly Interface:** A simple interface using **Streamlit** for recruiters to upload resumes and job descriptions.
- **Efficient Recruitment Process:** Reduce the time and effort required for manual screening of resumes.

1.4.2 Limitations:

While the project aims to automate resume ranking, certain constraints exist:

- **Contextual Understanding:** The system may not fully grasp subjective factors like cultural fit or soft skills.
- **Dependence on Data Quality:** Inaccurate or poorly formatted resumes may affect ranking accuracy.
- **Limited Industry-Specific Customization:** The model may need fine-tuning for specific industries with unique requirements.
- **No Final Hiring Decision:** The system provides rankings but does not replace human judgment in final hiring decisions.
- **Scalability Considerations:** Handling extremely large datasets may require additional computational resources.

CHAPTER 2

Literature Survey

2.1 Review of Relevant Literature

The process of resume screening and ranking has been a crucial aspect of recruitment, with multiple studies and technological advancements contributing to this domain. This section reviews existing research, methods, and tools that have been developed to automate and optimize resume ranking.

Traditional Resume Screening Methods

Earlier, recruiters manually reviewed resumes, which was a time-consuming and labor-intensive process. Some key challenges with traditional screening methods included:

- **Large Volume of Applications:** Hiring managers often receive hundreds of resumes for a single job posting.
- **Human Bias in Selection:** Unconscious biases may influence the screening process.
- **Inefficiency and Subjectivity:** The process lacks consistency, as different recruiters may evaluate resumes differently.

Automated Resume Screening Approaches

With the advancement of **Artificial Intelligence (AI)** and **Natural Language Processing (NLP)**, several techniques have been explored to automate resume ranking.

1. Rule-Based Systems

- Early attempts at automation relied on keyword-based filtering.
- Resumes were ranked based on the occurrence of specific words related to job descriptions.
- Limitation: These methods lacked semantic understanding and often led to inaccurate rankings.

2. Machine Learning-Based Approaches

- **Supervised Learning:** Models are trained using labeled datasets where resumes are pre-ranked by recruiters. Algorithms such as **Logistic Regression, Decision Trees, and Support Vector Machines (SVM)** have been used in earlier studies.
- **Unsupervised Learning:** Clustering techniques like **k-Means and Hierarchical Clustering** have been applied to categorize resumes based on similar attributes.

3. Natural Language Processing (NLP) for Resume Matching

- NLP techniques such as **TF-IDF, Word2Vec, BERT, and Named Entity Recognition (NER)** have been used to improve the understanding of job descriptions and resumes.

- Research has shown that **semantic similarity models** enhance ranking accuracy by considering context rather than just keyword matching.
4. **AI-Powered Resume Screening Tools**
- Several AI-driven Applicant Tracking Systems (ATS) have emerged, such as **LinkedIn Recruiter, HireVue, and iCIMS Talent Cloud**.
 - These tools integrate NLP and ML to automate screening but often require customization to align with different job roles.

Gap Analysis and Need for Improvement

While existing research and tools provide substantial improvements over manual screening, some limitations remain:

- Many **ATS systems rely heavily on keyword matching** rather than true semantic understanding.
- **Bias in AI models** due to training on historical hiring data.
- **Lack of explainability** in AI-based ranking systems, making it difficult to interpret ranking results.
- **Industry-Specific Challenges:** A generic model may not be equally effective for different domains like IT, healthcare, or finance.

2.2 Existing Models, Techniques, and Methodologies

Several models and methodologies have been developed to address the challenge of resume ranking and applicant screening. These approaches vary from traditional keyword-based filtering to advanced AI-driven models incorporating **Natural Language Processing (NLP) and Machine Learning (ML)** techniques. Below are some commonly used methods:

2.2.1 Comparison of Existing Models

- Reviewing different models, comparing **accuracy, methodology, pros/cons** in a table.
- Example:

Model	Technique Used	Accuracy	Limitations
TF-IDF	Statistical	85%	Ignores context
BERT	Deep Learning	92%	Computationally expensive

Comparison of Existing Models (Table – 1)

2.2.2 Traditional Keyword-Based Filtering

Overview:

- One of the earliest methods used in **Applicant Tracking Systems (ATS)**.
- Resumes are scanned for specific keywords that match the job description.

Techniques Used:

- **Boolean Search:** Uses logical operators (AND, OR, NOT) to filter resumes.
- **Regular Expressions:** Extracts relevant information based on patterns.
- **TF-IDF (Term Frequency - Inverse Document Frequency):** Assigns importance to words based on their frequency in a document relative to all documents.

Limitations:

- Lacks **semantic understanding** (e.g., “Software Engineer” vs. “Developer” may not be recognized as similar).
- Can be easily manipulated by **keyword stuffing** (adding keywords excessively to pass ATS filters).
- Ignores **context and quality of experience**.

2.2.3 Machine Learning-Based Models

Overview:

- Supervised and unsupervised learning models analyze and rank resumes based on training data.

Techniques Used:

- **Logistic Regression & Decision Trees:** Basic classification models used to rank resumes.
- **Support Vector Machines (SVM):** Classifies resumes into relevant and non-relevant categories based on job descriptions.
- **Random Forest & Gradient Boosting:** Advanced ensemble methods that improve ranking accuracy.

Limitations:

- Requires **labeled training data** (manually ranked resumes).
- Does not always generalize well to **new job roles and industries**.

2.2.4 NLP-Based Techniques for Resume Ranking

Overview:

- NLP enables **contextual analysis** by extracting meaning from text rather than relying on direct keyword matching.

Techniques Used:

- **Word2Vec & GloVe (Word Embeddings):** Represent words as numerical vectors based on their meaning.
- **BERT (Bidirectional Encoder Representations from Transformers):** Advanced deep learning model that understands context and improves ranking accuracy.
- **Named Entity Recognition (NER):** Extracts relevant details like **names, skills, education, and experience** from resumes.
- **Semantic Similarity Models (Cosine Similarity, Jaccard Similarity):** Measures how closely a resume matches a job description beyond just keywords.

Limitations:

- Requires **pre-trained models and fine-tuning** for specific industries.
- High **computational cost**, especially for deep learning models like BERT.

2.2.5 AI-Powered Resume Screening Systems

Overview:

- AI-driven **ATS platforms** use a combination of **ML and NLP** to rank resumes.

Examples of Existing AI Models:

- **LinkedIn Recruiter**: Uses AI to match job descriptions with potential candidates.
- **HireVue**: Incorporates AI-driven assessments along with resume screening.
- **iCIMS Talent Cloud**: Utilizes machine learning to rank applicants based on job fit.

Limitations:

- AI models may inherit **biases** from training data.
- Lacks **transparency**, making it difficult for recruiters to understand ranking decisions.

2.3 Gaps in Existing Solutions and How This Project Addresses Them

Despite the advancements in **resume ranking and applicant screening**, existing solutions still face several **limitations** that impact their effectiveness. Below are some common gaps in current methodologies and how this project aims to overcome them.

Gaps in Existing Solutions

2.3.1 Lack of Contextual Understanding

Issue:

- Traditional **keyword-based filtering** fails to understand the context of words.
- For example, a system may not recognize that “Software Engineer” and “Software Developer” are similar roles.

How Our Project Solves It:

- ✓ Implements **NLP-based semantic similarity models** to analyze the **meaning** of words rather than just matching keywords.
- ✓ Uses **BERT (Bidirectional Encoder Representations from Transformers)** or similar deep learning techniques to understand context.

2.3.2 Inability to Handle Variations in Resume Formats

Issue:

- Many systems struggle with **different resume structures** (e.g., PDFs, DOCX, or scanned images).
- Poor extraction of **skills, experience, and education** from non-standard formats.

How Our Project Solves It:

- ✓ Utilizes **Natural Language Processing (NLP)** techniques to extract structured data from unstructured resumes.
- ✓ Implements **Named Entity Recognition (NER)** to accurately identify candidate details across various formats.

2.3.3 Bias in AI-Based Resume Screening

Issue:

- AI models trained on biased datasets can **favor certain demographics, universities, or experience levels**.
- Lack of **explainability** in why a resume is ranked higher than another.

How Our Project Solves It:

- ✓ Uses **transparent ranking criteria** based on skillset relevance rather than hidden patterns in training data.
- ✓ Ensures a **fair evaluation** by focusing on skills, job descriptions, and qualifications without biased weighting.

2.3.4 Inefficiency in Matching Resumes with Job Descriptions**Issue:**

- Many resume ranking systems **do not accurately match job descriptions** with resumes due to reliance on simple similarity measures.
- Fails to capture the importance of **experience, projects, and certifications**.

How Our Project Solves It:

- ✓ Uses **advanced similarity metrics** like **cosine similarity** and **word embeddings** to better compare resumes with job descriptions.
- ✓ Incorporates **weighted scoring models** that prioritize experience, certifications, and relevant projects.

2.2.5 Lack of Real-Time Resume Ranking and User Interaction**Issue:**

- Most traditional systems do not provide **real-time feedback** on resume ranking.
- Candidates and recruiters cannot interact with the system dynamically.

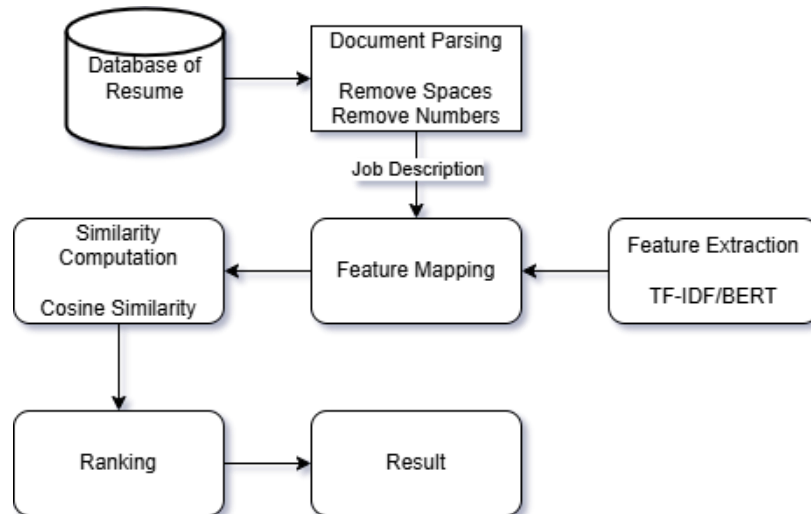
How Our Project Solves It:

- ✓ **Developed using Streamlit**, allowing an **interactive web interface** where users can upload resumes and get rankings instantly.
- ✓ Provides **real-time scoring and ranking** based on dynamically changing job descriptions.

CHAPTER 3

Proposed Methodology

3.1 System Design Explanation:



System Architecture (Figure – 1)

The system follows a structured pipeline to rank resumes based on a given job description. The key stages are:

1. **Database of Resumes**

- The system begins with a collection of resumes stored in a database. These resumes serve as input for further processing.

2. **Document Parsing**

- Each resume undergoes text preprocessing to remove unnecessary spaces, numbers, and unwanted characters.
- This step ensures that only relevant text data is used for feature extraction.

3. **Feature Extraction**

- The cleaned text data is converted into numerical representations using **TF-IDF (Term Frequency-Inverse Document Frequency)** or **BERT (Bidirectional Encoder Representations from Transformers)**.
- TF-IDF captures the importance of words in each resume relative to the entire dataset.
- BERT, a deep learning model, helps in understanding the semantic meaning of text.

4. **Feature Mapping**

- The job description is also preprocessed and converted into the same numerical representation as resumes (using TF-IDF/BERT).
- This ensures that resumes and job descriptions are comparable.

5. **Similarity Computation**

- Cosine Similarity is applied to measure how closely each resume matches the job description.
- A higher similarity score indicates a better match.

6. **Ranking**

- Resumes are ranked based on their similarity scores, with the top-ranked resumes being the most relevant to the job description.
- The system selects the **Top N** resumes that best match the job description.

7. **Final Result**

- The system outputs the best-matching resumes, helping recruiters quickly shortlist the most suitable candidates.

3.2 Requirement Specification:

This section outlines the tools and technologies required to implement the resume ranking system effectively. The requirements are categorized into **hardware** and **software** components to ensure smooth execution and optimal performance.

3.2.1 Hardware Requirements

The hardware requirements depend on the dataset size and the computational complexity of text processing and ranking. The recommended hardware includes:

- **Processor:** *Intel Core i5/i7 or AMD Ryzen 5/7 (or higher)*
 - A fast processor is necessary for handling large datasets and performing computations efficiently. A multi-core processor speeds up text processing tasks like document parsing and feature extraction.
- **RAM:** *Minimum 8GB (16GB recommended for better performance)*
 - Text processing and similarity computations can be memory-intensive, especially when dealing with thousands of resumes. Higher RAM ensures smooth execution without lagging or crashes.
- **Storage:** *Minimum 256GB SSD (512GB recommended for faster data processing)*
 - SSD (Solid State Drive) provides faster read/write speeds compared to HDD (Hard Disk Drive). Since resume data and models might require frequent file access, SSD storage significantly improves performance.
- **Graphics Processing Unit (GPU):** *Optional but recommended for deep learning models like BERT*
 - If the system integrates BERT (Bidirectional Encoder Representations from Transformers) for feature extraction, a GPU can accelerate model inference. However, for simpler TF-IDF-based implementations, a GPU is not necessary.

3.2.2 Software Requirements

To implement the resume ranking system, several software tools, libraries, and frameworks are required:

- **Operating System:** *Windows 10/11, Ubuntu (Linux), or macOS*
 - The project can run on any modern operating system. However, Linux-based environments (such as Ubuntu) are preferred for better compatibility with Python-based libraries and faster execution in production setups.
- **Programming Language:** *Python*
 - Python is chosen for its extensive ecosystem of libraries that support natural language processing (NLP), machine learning, and web development.
- **Frontend Framework:** *Streamlit (for interactive UI)*

- Streamlit is used to build a simple and interactive web-based interface for users to upload resumes and job descriptions. It enables quick visualization of ranked resumes without requiring advanced frontend development skills.
- **Backend:** *Python (FastAPI/Flask optional if required for expansion)*
 - While Streamlit handles the user interface, the backend logic can be extended using Flask or FastAPI if future API endpoints or database integrations are needed.
- **Libraries & Dependencies:**
 - **Text Processing:** *NLTK, spaCy*
 - These libraries help in preprocessing resumes and job descriptions by removing stopwords, tokenizing text, and performing lemmatization.
 - **Feature Extraction:** *Scikit-learn (TF-IDF), Transformers (BERT)*
 - TF-IDF (Term Frequency-Inverse Document Frequency) is a widely used technique for representing text as numerical vectors. BERT, a deep learning model, can be used for advanced semantic understanding.
 - **Similarity Computation:** *Scipy, Scikit-learn*
 - These libraries provide efficient implementations of cosine similarity, which is used to measure how closely a resume matches a job description.
 - **Machine Learning Model:** *K-Nearest Neighbors (KNN) (if ranking enhancement is needed)*
 - KNN can be used for ranking resumes by finding the top 10 closest resumes to a given job description. It enhances the ranking system by leveraging distance-based similarity.


CHAPTER 4

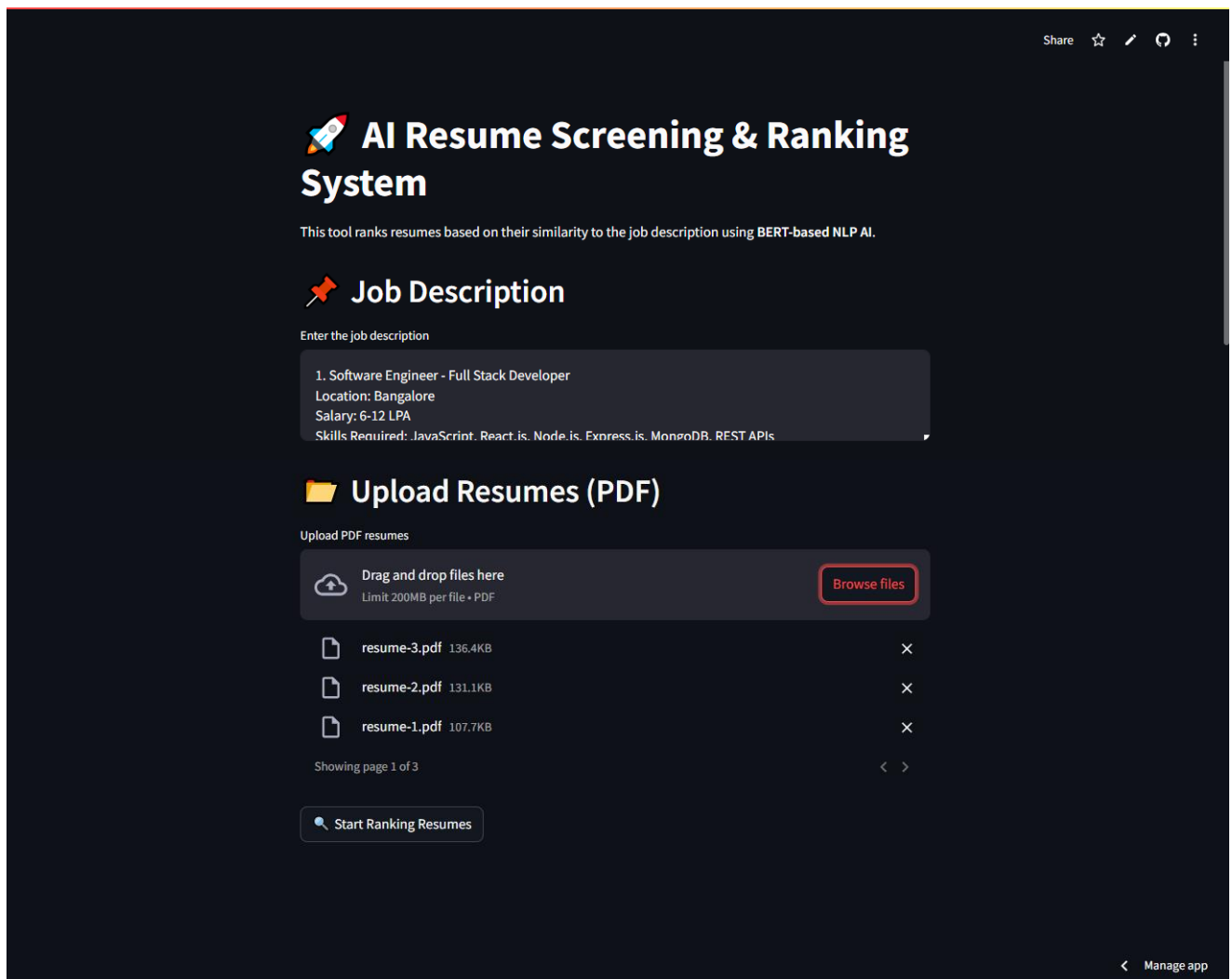
Implementation and Result

4.1 Snapshot of the Project:

4.1.1 Uploading Resumes & Job Description (Input Snapshot)

 **Snapshot:** A screenshot of the **Streamlit UI** where users can upload multiple resumes (PDF/DOCX) and enter a job description.

 **Explanation:** This snapshot demonstrates the input phase, where users upload resumes and enter a job description to get the top matching candidates. It highlights the simplicity and user-friendliness of the interface.



Input Snapshot (Figure – 2)

4.2 Github Link for the Project:

GitHub Repository: https://github.com/rajesh31082001/Resume_Ranking

CHAPTER 5

Discussion and Conclusion

5.1 Future Work

While the current resume ranking system efficiently matches resumes to job descriptions using **TF-IDF/BERT and cosine similarity**, there are several ways it can be enhanced to improve accuracy, usability, and scalability.

1. Enhancing NLP Capabilities

- **Incorporating Deep Learning Models:** Instead of relying solely on TF-IDF/BERT, advanced transformer-based models such as **RoBERTa, XLNet, or GPT** could be used to improve text understanding and ranking accuracy.
- **Contextual Keyword Matching:** Implementing **Named Entity Recognition (NER)** to better extract and match domain-specific skills, experiences, and qualifications from resumes.
- **Handling Synonyms & Job Title Variations:** Using **word embeddings (FastText, Word2Vec, GloVe)** to improve similarity detection by understanding words in different contexts.

2. Expanding File Format Support

- **Multi-Format Resume Parsing:** Extending support for **PDF, DOCX, and scanned image-based resumes** using **OCR (Optical Character Recognition)** techniques.
- **Better Resume Structuring:** Implementing **semantic segmentation** to identify and extract key sections (e.g., Education, Experience, Skills) more effectively.

3. Machine Learning-Based Ranking

- **Supervised Learning Approaches:** Training a machine learning model (**Random Forest, XGBoost, or Deep Learning**) on labeled hiring data to predict the best resume rankings.
- **Reinforcement Learning for Adaptive Ranking:** Using feedback from recruiters and hiring trends to improve the ranking algorithm dynamically over time.

4. Industry-Specific Customization

- **Domain-Specific Model Fine-Tuning:** Creating **custom models for different industries (IT, Healthcare, Finance, etc.)** to improve relevance.
- **Customizable Ranking Weights:** Allowing recruiters to adjust ranking weights based on experience, skills, or certifications required for a job.

5. Real-Time Processing & API Integration

- **Developing a Web-Based Platform:** Converting the system into a **full-fledged SaaS** where recruiters can upload job descriptions and resumes for **real-time ranking**.
- **API for ATS (Applicant Tracking System) Integration:** Providing an API for HR software to automatically process and rank resumes within hiring platforms.

6. Scalability and Performance Optimization

- **Parallel Processing for Faster Ranking:** Implementing **multi-threading or distributed computing** to handle large-scale resume databases efficiently.
- **Cloud Deployment:** Deploying the system on **AWS, Azure, or Google Cloud** to make it scalable and accessible for enterprise use.

5.2 Conclusion

The **AI-driven Resume Ranking System** effectively addresses the challenge of filtering and ranking resumes based on job descriptions by leveraging **Natural Language Processing (NLP) techniques** like **TF-IDF/BERT and Cosine Similarity**. This project enhances the hiring process by automating candidate shortlisting, reducing manual effort, and improving the accuracy of resume screening.

Through **document parsing, feature extraction, and similarity computation**, the system provides a ranked list of the most relevant resumes, ensuring that recruiters can quickly identify the best candidates. The integration of **advanced text-processing techniques** improves the system's efficiency in handling large datasets, making it a **valuable tool for HR professionals and recruiters**.

While the current model provides significant improvements over traditional resume screening methods, future enhancements—including **deep learning integration, domain-specific fine-tuning, real-time API deployment, and cloud-based scalability**—can further increase its effectiveness. The project serves as a **foundation for AI-powered recruitment tools** and demonstrates the potential of machine learning in **streamlining hiring processes**.

Overall, this system **contributes to the field of AI and HR tech** by offering an **automated, efficient, and scalable** solution to resume ranking, paving the way for more sophisticated recruitment technologies in the future.

REFERENCES

- [1].Ming-Hsuan Yang, David J. Kriegman, Narendra Ahuja, “Detecting Faces in Images: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24, No. 1, 2002.
- [2].Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.
- [3]. Ramos, J. (2003). "Using TF-IDF to Determine Word Relevance in Document Queries." *Proceedings of the First International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*.
- [4]. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). “Distributed Representations of Words and Phrases and Their Compositionality.” *Advances in Neural Information Processing Systems (NeurIPS)*.
- [5]. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” *Proceedings of the NAACL-HLT 2019*.
- [6]. Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
- [7]. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). “Scikit-learn: Machine Learning in Python.” *Journal of Machine Learning Research*, 12, 2825-2830.
- [8]. Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features." *Proceedings of the European Conference on Machine Learning (ECML)*.
- [9]. Cover, T., & Hart, P. (1967). “Nearest Neighbor Pattern Classification.” *IEEE Transactions on Information Theory*, 13(1), 21-27.
- [10]. Aggarwal, C. C., & Zhai, C. (2012). *Mining Text Data*. Springer.