Resume Analysis Tool

DATS 6312: Natural Language Processing
Fall 2023

Instructor: Dr. Amir Jafari

Individual Final Project Report

Group 1

Jadhav, Shubham
G30570862
Department of Computer Science
School of Engineering
George Washington University

GitHub Repository: https://github.com/JahnaviRamagiri/Final-Project-Group1

December 11, 2023

Abstract

This project aimed to improve the efficiency of hiring by automatically classifying resumes into the most suitable job roles. We compared various models, from simple tools like Naive Bayes to advanced transformer-based models like BERT using PyTorch. One unique aspect of our work was a resume-to-job description comparison. This "similarity score" helps measure how well a resume matches a specific job posting. Additionally, we developed a system to automatically generate informative summaries of resumes.

By comparing these different models, we gained valuable insights into how to best analyze resumes. Our work not only helps companies find the right candidates for their jobs but also provides valuable information for researchers in the field of natural language processing.

Keywords—Resume, Train, Test, Loss, Classification, Transformer

Contents

List of Figures

List of Tables

1 Introduction

The primary motivation for this project is to address the challenges candidates face when applying for job roles. Every candidate goes through the time-consuming process of shortlisting job postings based on their resumes manually. The difficulty in accurately understanding if they or their resume is well suited for the job requirement is alarming.

The Resume Analysis tool, aims to simplify and expedite the job application process, benefiting both candidates and also recruiters (future extension). Through the use of advanced NLP techniques, it seeks to provide more accurate and efficient resume analysis and job role matching. This aims to streamline this process by leveraging NLP techniques to achieve the following objectives:

1. **Resume - Job Description Similarity Score**: Calculate the similarity score between a candidate's resume and a job description to assess how well the resume matches the job requirements.
2. **Multi-class Resume Classification**: Categorize resumes into predefined classes or job roles, making it easier for candidates to identify suitable job roles for their resume.
3. **Resume Analysis**: Provide comprehensive analysis and insights into a candidate's resume, including key skills, qualifications, and experience.

2 Dataset

Source Link: https://github.com/florex/resume_corpus/tree/master

The multi-labeled dataset of resumes labeled with occupations. The resume files have the extension ".txt" and the corresponding labels are in a file with the extension ".lab".

This dataset contains 3 files :

1. resumes_corpus.zip: This file contains a set of resumes files with the extension ".txt" with the corresponding list of labels in a file with the extension ".lab"
2. resumes_sample.zip : This file represents the data set of resumes in a single text file. Each line of the file contains information about a text resume. Each line has 3 fields separated by ":::". The first field is the reference id of the resume; the second field is the list of occupations separated by ";" ; and the third field is the text resume.
3. normalized_classes : This file contains the associations between the occupations as written by the experts and their corresponding normalized form.

3 Description of Individual Work

Individually, my main area of focus were as follows:

1. **Tool Features**: Actively participated in brainstorming sessions to identify and finalize features for the Resume Scanner Tool.
2. **Data Gathering**: Performed data accumulation and extracted usable data the raw files into pandas data frame.
3. **Data Cleaning of resumes**: Created clean_resume_text method processes individual resume texts.
4. **Resume Summarization**
5. **Resume-Job Description Similarity**
6. **Resume-Job Role Multi Label Classification**
   (a) **Multinomial Naive Bayes Classifier**
   (b) **BERT with Linear head**
   (c) **Hyperparameter Tuning**
7. **Streamlit function integration**

3.1 Architecture

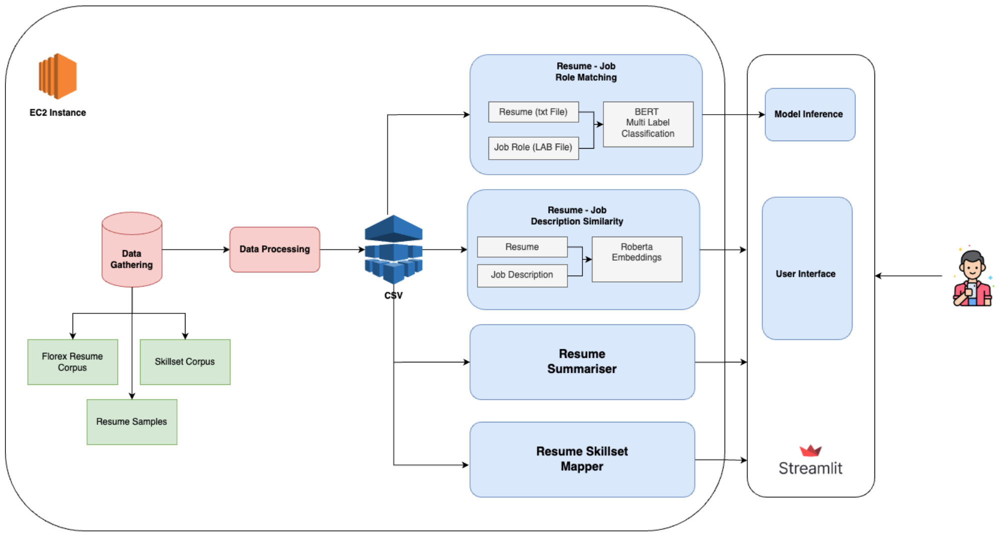


Figure 1: Model Architecture of proposed system

3.2 Data Preprocessing

Data preprocessing is a crucial step in the data analysis and machine learning pipeline. It involves cleaning and transforming raw data into a format that is suitable for analysis or for training machine learning models. The quality of the data and the features extracted from it significantly impact the performance of any data-driven project.

3.2.1 Data Gathering

The initial phase of the project starts with accumulating all the available data. From section 2 we can see that we have 2 zip files and one normalized classes text file. The objective here is to create dataset in pandas which are easier to manipulate for various tasks. The resume_corpus file has two type of files i.e., txt file with resume and lbl file with job roles for the corresponding resume. We iterate a loop over all the files and create a dataset with two columns. First column has resume text and second column with job roles seperated with comma. The resume_samples file is a single text file with all the resume and corresponding skillset in each sentence for each resume. We import this file in python and loop over sentence and extract resumes and skillset which are seperated by ":::". All the files are stored as Uncleaned dataset.

3.2.2 Data Cleaning

Once the data is curated and placed in the Uncleaned Folder, the next step is to clean it and process it for the next models. Data Cleaning is mainly done in on Resumes and SkillSet.

Resume: The clean_resume_text method processes individual resume texts by:

1. Removing HTML tags using BeautifulSoup.
2. Eliminating special characters and punctuation with regular expressions.
3. Lowercasing all text for consistency.
4. Tokenizing the text into individual words.
5. Lemmatizing tokens for standardization.
6. Removing duplicate neighboring tokens.
7. Eliminating common English stopwords.
8. Concatenating cleaned tokens into a single string for further analysis.

3.3 Resume Summarization

In our project, we implemented a text summarization approach utilizing the **Hugging Face Transformers** library, specifically employing the "summarization" pipeline. The initial step involved extracting textual information from the resumes, followed by a basic data cleaning process to enhance the quality of the input data. Leveraging the transformer-based **BART (Bidirectional and Auto-Regressive Transformers)** model, the summarizer was applied to generate coherent and relevant summaries [1]. The integration of BART ensured the effectiveness of our summarization task, as it is specifically tailored for sequence-to-sequence tasks, such as summarization and text generation. The flexibility of the Hugging Face pipeline allowed us to fine-tune summarization parameters, such as **max_length** to optimize the summarization results [2].

3.4 Resume-Job Description Similarity

We implemented two distinct approaches to enhance the analysis of resumes in relation to job descriptions. The first method utilized TF-IDF (Term Frequency-Inverse Document Frequency) Cosine Similarity, a traditional yet effective technique for representing document content and measuring the semantic similarity between documents [3, 4].

In addition to TF-IDF, we incorporated state-of-the-art transformer-based models, namely BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa (Robustly optimized BERT approach) [5, 6]. For both the resume and job description, we tokenized the text and computed embeddings using BERT and RoBERTa Tokenizer. To obtain document-level embeddings, we employed mean pooling, providing a simplified representation of the texts. Lastly, we calculated the cosine similarity between the embeddings generated by BERT and RoBERTa between resume and job description.

The cosine similarity scores obtained from BERT and RoBERTa embeddings offer a more advanced and context-aware measure of similarity compared to the TF-IDF method.

$$\cos(\theta) = \frac{Resume \cdot JobDescription}{\|Resume\|_2 \, \|JobDescription\|_2}$$

3.5 Multi Label Classification

The primary goal of the resume analysis tool is to perform multilabel classification of resumes into suitable job roles. Given that a single resume can align with multiple job roles, this task falls under the category of multilabel classification. The dataset, resume_corpus, contains a mapping between .txt and .lab files, where the former holds the resume text, and the latter specifies the associated job roles. To kickstart this module, we opt for Multinomial Naive Bayes (MultinomialNB) as our foundational model. As a next step, we explore the implementation of a BERT Base model. This approach allows us to evaluate and compare the performance of both models, ensuring a comprehensive understanding of their effectiveness in addressing the multilabel classification challenge in the context of resume analysis.

3.5.1 Multinomial Naive Bayes Classifier

The Multinomial Naive Bayes (MultinomialNB) classifier is a powerful algorithm known for its simplicity and effectiveness in text classification tasks [7]. It is a probabilistic algorithm based on Bayes' theorem, with an assumption of independence between features.

To establish a baseline for comparison with state-of-the-art models, we have chosen to implement the Multinomial Naive Bayes (MultinomialNB) classifier. This classifier is renowned for its simplicity and effectiveness, making it a formidable candidate for text classification tasks, especially when dealing with discrete data such as word counts in documents [7]. The MultinomialNB classifier operates on the principles of Bayes' theorem, leveraging a probabilistic approach with an assumption of feature independence.

By employing the MultinomialNB classifier, we aim to assess its performance as a foundational model in the complex task of classifying resumes into multiple job roles. This classifier's simplicity and reliance on probabilistic reasoning make it an excellent candidate for comparison against more advanced models. The intention is to use the MultinomialNB classifier as a baseline to gauge the efficacy of state-of-the-art models, thus providing a valuable reference point in evaluating the advancements and nuances introduced by more sophisticated classifiers in the realm of multi-label resume classification.

$$P(A|B) = P(A) * \frac{P(B|A)}{P(B)}$$

3.5.2 BERT+Linear

In the initial model architecture, BERT is coupled with a linear classifier for multiclass classification. This configuration represents a straightforward approach, leveraging the contextualized embeddings from BERT and connecting them to a linear layer that outputs class probabilities.

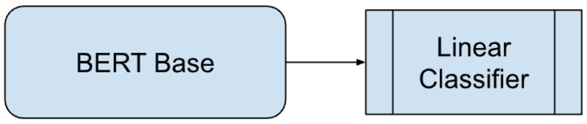


Figure 2: BERT with Linear head

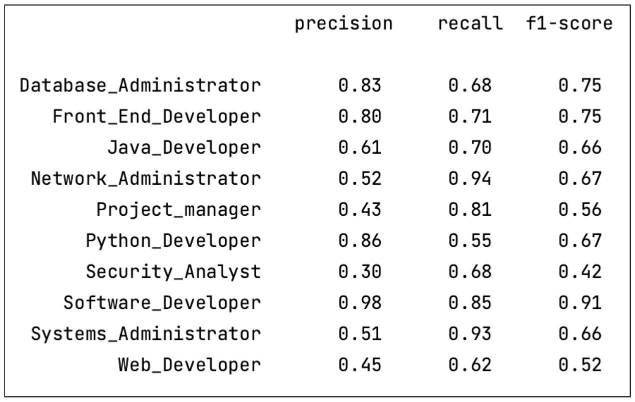
In the BERT + Linear Classifier configuration, the BERT model is employed to generate contextualized word embeddings for each token in the input sequence. These embeddings encapsulate the semantic meaning of words in the context of the entire document. The resulting embeddings are then flattened or pooled to create a fixed-size representation of the entire sequence. This representation is then passed through a linear layer, which acts as a classifier. The linear layer maps the BERT embeddings to the number of output classes, producing a probability distribution over these classes. During training, the model learns the optimal weights for the linear layer by minimizing the Binary cross-entropy logits loss between predicted probabilities and actual labels.

3.6 Model Optimization

1. **Learning Rate**: We experimented with a range from 0.001 to 2e-5.
2. **Batch Size**: We experimented with a range from 16 to 32 to identify the optimal balance for our model's memory constraints and learning stability.
3. **Epochs**: We restricted the number of epochs to a range of 3 to 10 to balance between underfitting and overfitting.
4. **Max Length**: The sequence length varied within the range of 64 to 256 tokens, enabling effective management of the context window.

3.7 Results

3.7.1 Multinomial Naive Bayes Classifier



```
                           precision    recall  f1-score


Database_Administrator       0.83      0.68      0.75
   Front_End_Developer       0.80      0.71      0.75
        Java_Developer       0.61      0.70      0.66
 Network_Administrator       0.52      0.94      0.67
       Project_manager       0.43      0.81      0.56
      Python_Developer       0.86      0.55      0.67
      Security_Analyst       0.30      0.68      0.42
    Software_Developer       0.98      0.85      0.91
 Systems_Administrator       0.51      0.93      0.66
         Web_Developer       0.45      0.62      0.52
```

Figure 3: Classification Report from Multinomial Naive Bayes Classifier

We can observe that the f1-score across all classes is approximately the same and in the range of 0.65 to 0.75 expect for Security Analyst, Project Manager, and Web Developer. The conclusive performance mertrics for the Multinomial Naive Bayes Classifier highlights is limited efficacy with a Precision of 0.54, Recall of 0.85, and F1 Score of 0.66.

3.7.2 BERT + Linear

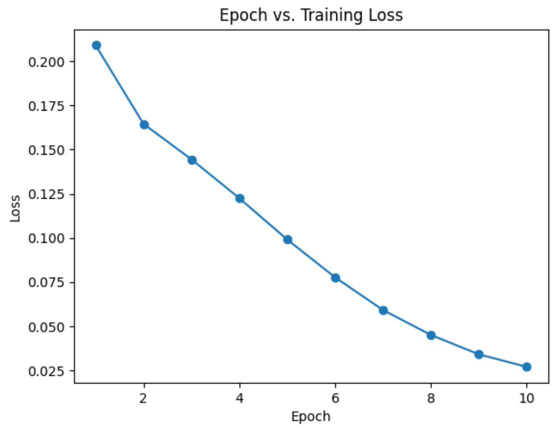


Figure 4: Plot of training loss across epoch

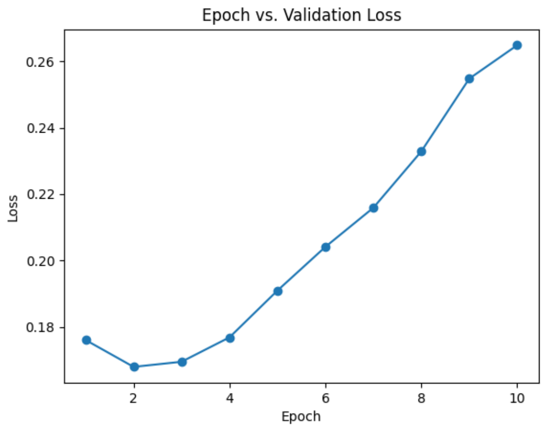
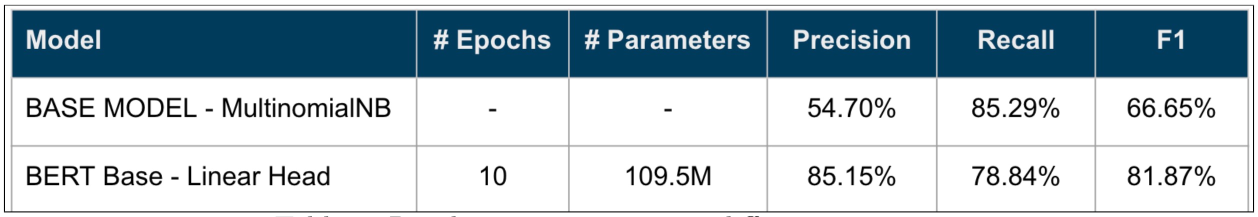


Figure 5: Plot of validation loss across epoch

The BERT-Linear classifier was trained over 10 epochs, as illustrated by the Training and Validation Loss figure. During training, the loss consistently decreased, reflecting effective learning. However, the validation loss exhibited a gradual increase after the 2nd epoch, suggesting potential overfitting. Following a thorough analysis, Epoch 2 was identified as the optimal stopping point, ensuring a balance between learning and generalization.
The conclusive performance metrics for the optimal BERT-Linear model showcase its efficacy. With a Precision of 0.85, Recall of 0.78, and F1 Score of 0.81

3.7.3 Model Comparison



Model	# Epochs	# Parameters	Precision	Recall	F1
BASE MODEL - MultinomialNB	-	-	54.70%	85.29%	66.65%
BERT Base - Linear Head	10	109.5M	85.15%	78.84%	81.87%

Table 1: Result comparison across different experiments

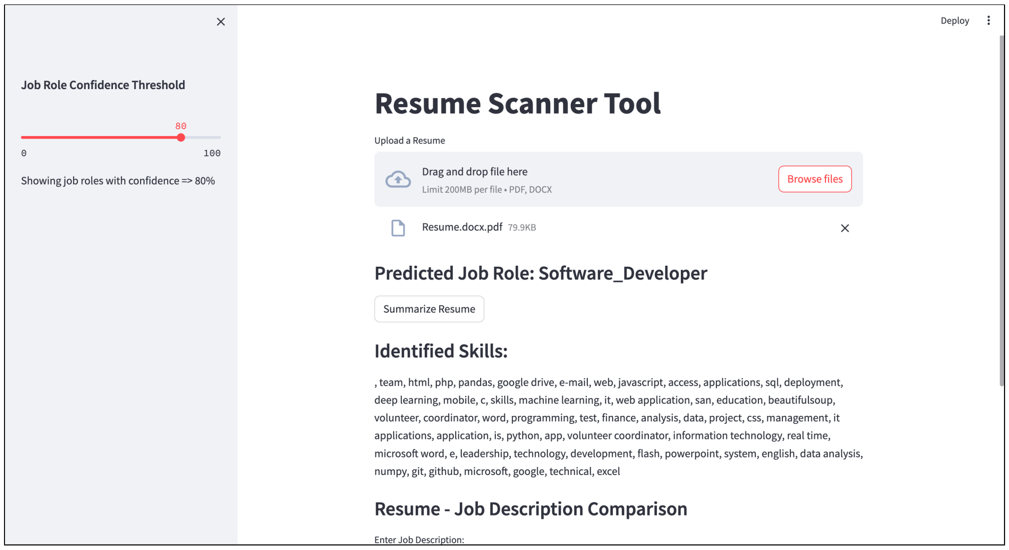
From the table we can see that, BERT Base - Linear outperforms the Base Model i.e., MulinomialNB with a high Precision and Recall value of 85.15%. The optimised model has an overall F1 of 81.87%.

3.8 Streamlit Application

The Resume Scanner Tool is developed using Streamlit, a powerful Python library for creating interactive web applications with minimal code. The tool allows users to upload resumes in PDF or DOCX format, leveraging helper functions from helperfunctions.py for parsing, classification, summarization, skillset identification, and job description comparison.

I have worked on establishing the functionality required to connect NLP results to the streamlit UI. I have also been involved in brain storming the design and setting up boiler plate for the application. Following are the sections that I have worked on:

1. Method to parse the uploaded pdf resume into text format
2. Method to performing basic pre-processing on the text resume
3. Method to summarize the text resume
4. Iniatize the model and classify the input resume into job roles
5. Integrate slider value input with prediction function
6. Method to find similarity between job description and resume


Job Role Confidence Threshold

80

0 100

Showing job roles with confidence => 80%

Deploy

Resume Scanner Tool

Upload a Resume

Drag and drop file here
Limit 200MB per file • PDF, DOCX

Browse files

Resume.docx.pdf 79.9KB

Predicted Job Role: Software_Developer

Summarize Resume

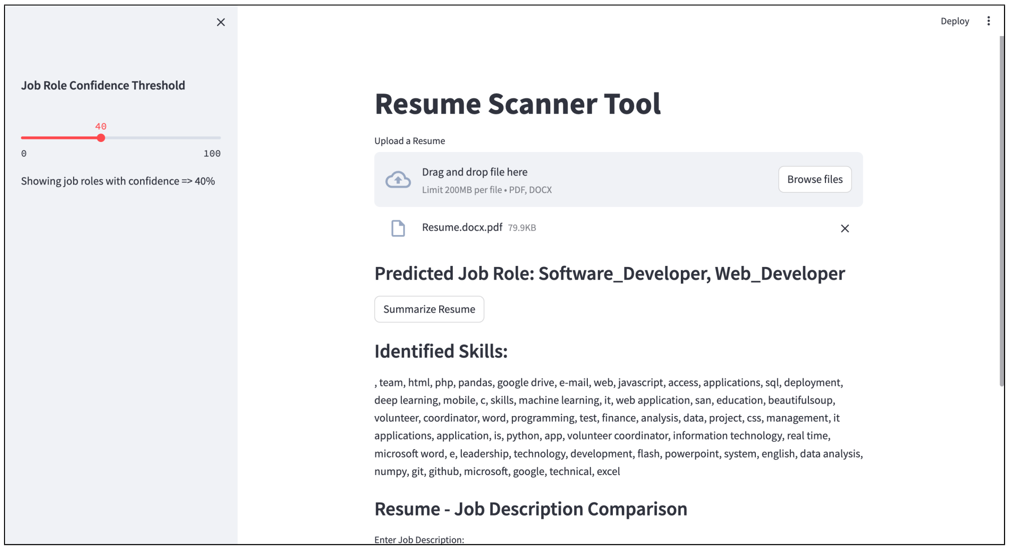
Identified Skills:

, team, html, php, pandas, google drive, e-mail, web, javascript, access, applications, sql, deployment, deep learning, mobile, c, skills, machine learning, it, web application, san, education, beautifulsoup, volunteer, coordinator, word, programming, test, finance, analysis, data, project, css, management, it applications, application, is, python, app, volunteer coordinator, information technology, real time, microsoft word, e, leadership, technology, development, flash, powerpoint, system, english, data analysis, numpy, git, github, microsoft, google, technical, excel

Resume - Job Description Comparison

Enter Job Description:

Figure 6: Predicting job roles from resume


Job Role Confidence Threshold

40

0 100

Showing job roles with confidence => 40%

Deploy

Resume Scanner Tool

Upload a Resume

Drag and drop file here
Limit 200MB per file • PDF, DOCX

Browse files

Resume.docx.pdf 79.9KB

Predicted Job Role: Software_Developer, Web_Developer

Summarize Resume

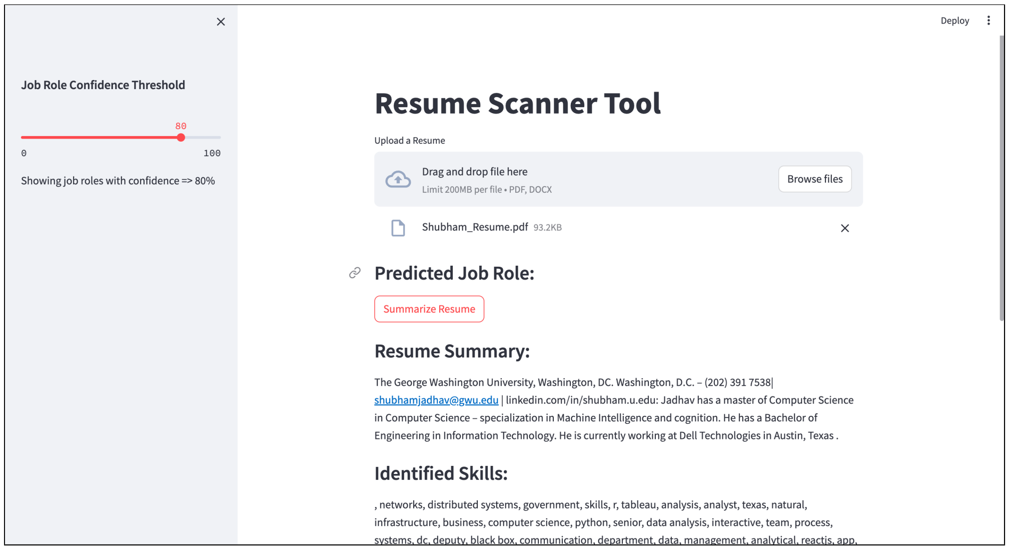
Identified Skills:

, team, html, php, pandas, google drive, e-mail, web, javascript, access, applications, sql, deployment, deep learning, mobile, c, skills, machine learning, it, web application, san, education, beautifulsoup, volunteer, coordinator, word, programming, test, finance, analysis, data, project, css, management, it applications, application, is, python, app, volunteer coordinator, information technology, real time, microsoft word, e, leadership, technology, development, flash, powerpoint, system, english, data analysis, numpy, git, github, microsoft, google, technical, excel

Resume - Job Description Comparison

Enter Job Description:

Figure 7: Predicted Job Roles with 45% threshold



Showing job roles with confidence => 80%

Resume Scanner Tool

Upload a Resume

Limit 200MB per file • PDF, DOCX

Shubham_Resume.pdf 93.2KB

Predicted Job Role:

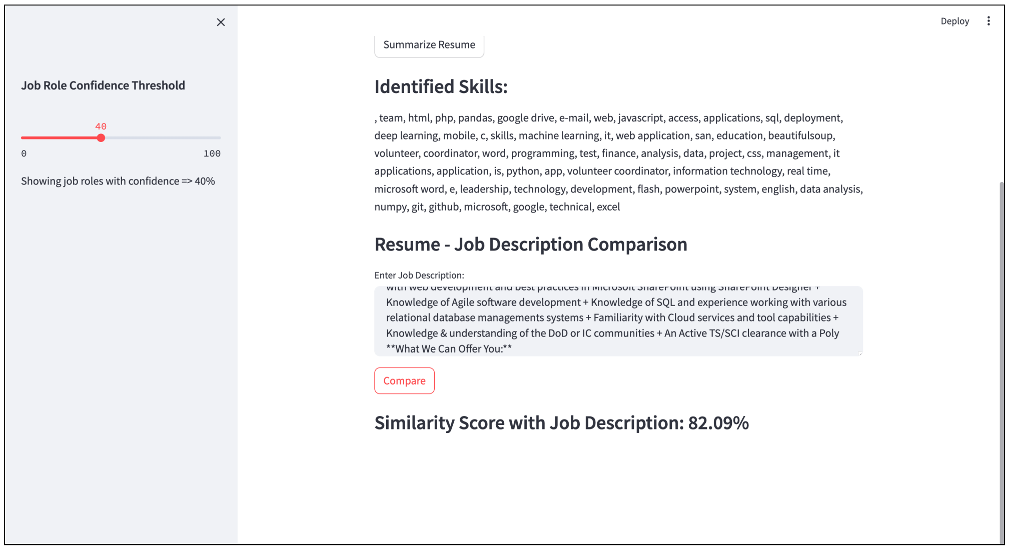
Resume Summary:

The George Washington University, Washington, DC. Washington, D.C. – (202) 391 7538| shubhamjadhav@gwu.edu | linkedin.com/in/shubham.u.edu: Jadhav has a master of Computer Science in Computer Science – specialization in Machine Intelligence and cognition. He has a Bachelor of Engineering in Information Technology. He is currently working at Dell Technologies in Austin, Texas .

Identified Skills:

, networks, distributed systems, government, skills, r, tableau, analysis, analyst, texas, natural, infrastructure, business, computer science, python, senior, data analysis, interactive, team, process, systems, dc, deputy, black box, communication, department, data, management, analytical, reactjs, app,

Figure 8: Displaying Resume summary on demand



Showing job roles with confidence => 40%

Identified Skills:

, team, html, php, pandas, google drive, e-mail, web, javascript, access, applications, sql, deployment, deep learning, mobile, c, skills, machine learning, it, web application, san, education, beautifulsoup, volunteer, coordinator, word, programming, test, finance, analysis, data, project, css, management, it applications, application, is, python, app, volunteer coordinator, information technology, real time, microsoft word, e, leadership, technology, development, flash, powerpoint, system, english, data analysis, numpy, git, github, microsoft, google, technical, excel

Resume - Job Description Comparison

Enter Job Description:

with web development and best practices in Microsoft SharePoint using SharePoint Designer + Knowledge of Agile software development + Knowledge of SQL and experience working with various relational database managements systems + Familiarity with Cloud services and tool capabilities + Knowledge & understanding of the DoD or IC communities + An Active TS/SCI clearance with a Poly **What We Can Offer You:**

Similarity Score with Job Description: 82.09%

Figure 9: Displaying similarity score of resume and job description

4 Conclusion

In summary, this project focuses on enhancing the efficiency of the job application process through the development of the Resume Analysis tool. My key contributions include the implementation of data gathering and resume clearning process. Creating Base model of Naive Bayes and state of the art NLP systems such as BERT models with Linear classifier heads, for accurate job role classification and resume analysis. Additionally, resume summary extraction and resume to job description similarity provides a detailed oriented platform for candidates and recruiter to assist them in recruiting process.

The backend integration for Streamlit-based UI were developed to provide a path to use trained model to be usable with easy to interact tool for all users. The results demonstrate the effectiveness of the BERT-Linear model over the base model. The implementation of BERT model laid down the base to build ensemble models with difference heads such as CNN and LSTM, which in turn resulted in better performance. Overall, the project successfully addresses the challenges in resume analysis, providing a valuable resource for optimizing the job application experience.

5 Future Scope

5.1 Interpretable Models

Focusing on developing interpretable models by employing techniques such as attention visualization and saliency maps. This enhances transparency and allows users to understand how the model arrives at specific predictions.

5.2 Optical character recognition

We can extend the initial phase of parsing pdf or docx resume to extracting text from resume which are in a image format.

5.3 Automated Feedback Loop

Establishing an automated feedback loop for continuous improvement, where user feedback on model predictions contributes to ongoing model refinement. This process will help train a better model.

5.4 Multimodal Resume Analysis

Incorporating multimodal analysis by considering not only textual information but also visual elements from resumes, such as images, graphics, and portfolios. This could provide a more holistic understanding of a candidate's qualifications.

6 Code Contribution

According to the formula, the code percentage is **38%**.

References

[1] https://huggingface.co/docs/transformers/model_doc/bart.

[2] A. Jafari. https://github.com/amir-jafari/NLP/tree/master/Lecture_09.

[3] https://huggingface.co/tasks/sentence-similarity.

[4] A. Jafari. https://github.com/amir-jafari/NLP/tree/master/Lecture_04.

[5] https://huggingface.co/docs/transformers/model_doc/bert.

[6] https://huggingface.co/docs/transformers/model_doc/roberta.

[7] E. Gomede, "Understanding multinomial naive bayes classifier." https://medium.com/@evertongomede/understanding-multinomial-naive-bayes-classifier-fdbd41b405bf, Nov 2011.