CS F320 - Foundation of Data Science

Resume ATS Scorer and Matcher : A Complete Report

Submitted to
Dr. Tejasvi Alladi

On
23rd April, 2025

By Group Members

Name	BITS ID	Email;
Vedant Vyas	2021A7PS2693P	f20212693@pilani.bits-pilani.ac.in
Nachiketsingh Kandari	2021A7PS2691P	f20212691@pilani.bits-pilani.ac.in
Samarth Gandotra	2021A7PS2437P	f20212437@pilani.bits-pilani.ac.in
Amandeep Singh	2021A7PS0575P	f20210575@pilani.bits-pilani.ac.in

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

Abstract

The Resume ATS Scorer and Matcher helps to address a critical challenge in modern job seeking: optimizing resumes to pass through Applicant Tracking Systems (ATS). The Applicant Tracking System or ATS is used by 99% of Fortune 500 companies and 75% of mid-sized organizations.

This system combines deep semantic analysis through BERT (Bidirectional Encoder Representations from Transformers) embeddings with traditional TF-IDF (Term Frequency-Inverse Document Frequency) keyword matching to evaluate resumes against job descriptions.

By parsing PDF resumes into structured components (education, experience, skills), it calculates both lexical matches and understands context for relevance and provides a dual-layered evaluation which is close to how human recruiters and ATS systems assess candidates.

This project will give a score for the resume from 0 to 100% along with detailed feedback on keyword gaps, skill relevance and semantic alignment. This will enable job seekers to optimize their resumes for the job titles.

Some unique features of this project include batch processing for multiple applications and extended analysis for career path optimization.This feature addresses both immediate job applications and long-term career development needs.

Problem Statement

Context and Industry Challenge

The current job market has become very dependent on ATS platforms, with 98.2% of Fortune 500 companies and 70% of medium sized companies using these systems in order to filter resumes. However, studies show that 75% of qualified applicants are rejected at the ATS stage. This happens due to resume formatting and keyword mismatch. This creates a barrier for job seekers who often do not know how ATS algorithms will evaluate their resumes.

Key Challenges Addressed

1. **Keyword Optimization Paradox:** Job seekers must balance keyword density (for ATS) with natural language (for human readers) . Traditional tools focus solely on keyword matching, leading to over optimised resumes that score well in ATS but perform poorly in human evaluations.
2. **Semantic Understanding Gap:** Current ATS systems do not understand the semantic meaning of the skills put up by the job applicants .For example, a resume which has the keyword "TensorFlow" in it might not match a job description requiring "deep learning frameworks" despite being qualified.
3. **Lack of Personalized Feedback:** Existing resume checkers provide generic suggestions ("add more keywords") without job-specific guidance or actionable improvement metrics.
4. **Time-Consuming Manual Tailoring:** Professionals applying to 10-15 positions weekly spend 6 to 8 hours manually customizing resumes, often without understanding which changes impact ATS scoring.
5. **Career Progression Planning:** Job seekers currently lack tools to evaluate their resume for career paths or roles they could qualify for with minor modifications.

Limitations of Current Solutions

Existing approaches fail to address these challenges comprehensively:

Approach	**Limitations**
Basic ATS Checkers	These only count keyword matches without any semantic understanding
Professional Services	Are expensive and only provide with static feedback
Generic Resume Advice	Not tailored to specific roles/industries. The advice given is general in nature
Manual Optimization	Time-intensive with unpredictable results since one doesn't know what to change.

Project Goals

The primary objectives of the Resume ATS Scorer and Matcher project are:

1. To parse and extract relevant information from PDF resumes with high accuracy
2. To perform semantic matching between resume content and job descriptions using BERT embeddings
3. To identify keyword matches and gaps using TF-IDF analysis
4. To calculate a comprehensive match score between resumes and job descriptions
   To provide actionable recommendations for resume improvement
5. To support batch processing of multiple resumes against various job descriptions

Key Technologies

Natural Language Processing Frameworks

BERT (Bidirectional Encoder Representations from Transformers)
BERT is an open-source natural language processing machine learning library developed by Google AI Language in 2018. Unlike the traditional language models that process text sequentially (left-to-right or right-to-left), BERT uses a bidirectional approach that considers both left and right context simultaneously. This enables BERT to construct a much deeper sense of language context and nuance.
BERT uses an ensemble-only transformer architecture that focuses on translating input sentences instead of forecasting output sentences. It's pre-trained on large text corpora using two new old strategies:

- **Masked Language Model (MLM):** Randomly masking words and the model having to guess original values based on context in sentence.
- **Next Sentence Prediction (NSP)**: Preparing the model to understand sentence-to-sentence relations

This pre-training generates a universal model that can be fine-tuned to specific NLP tasks with minimal or no modification and serves as a "swiss army knife" for language processing tasks such as sentiment analysis, named entity recognition, and question answering.

TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF is a quantitative measure to find the importance of terms in a document relative to a collection of documents (corpus). Unlike simple word frequency, TF-IDF scales down the frequency of frequent and rare words to identify the most important words in documents.
It combines two components:

- **Term Frequency (TF)**: Identifies the frequency of how often a term appears in a document and suggests its importance to the document
- **Inverse Document Frequency (IDF)**: Identifies the frequency of how often a term appears in a document and suggests its importance to the document

The formula uses logarithmic scaling to minimize the effect of very large or very small values so that scores scale well over diverse sets of documents. The statistical technique is applied to identify distinctive keywords that describe document content and rule out common words that add little meaning.

spaCy

spaCy is an open source, free Python package for high-performance natural language processing created by Ines Montani and Matthew Honnibal in 2015. spaCy is intended to process a lot of text efficiently and therefore well-suited for production use.
spaCy comes with industrial-strength NLP components like:

- Named entity recognition
- Part-of-speech tagging
- Dependency parsing
- Noun phrase extraction
- Text classification
- Tokenization and sentence segmentation

It is also integrated with pre-trained models in several languages and includes ease of integration with deep learning frameworks. SpaCy, version 3.0, had major improvements with transformer-based pipelines for better accuracy, wider configuration possibilities, and parallelized/distributed modes for faster rounds of training.

NLTK (Natural Language Toolkit)

NLTK is a free Python natural language processing library, developed by Steven Bird and Edward Loper in 2001. NLTK was initially programmed as an instructional tool for linguistic concepts and has improved and evolved into a full-featured set of text analysis tools.
NLTK provides robust tools for:

- Text tokenization (breaking text into words or sentences)
- Word stemming and lemmatization
- Part-of-speech tagging
- Named entity recognition
- Sentiment analysis
- Parsing and semantic reasoning

The library includes extensive corpora and lexical resources that help researchers in exploring patterns of language and making NLP models. The detailed documentation of NLTK makes it an extremely useful tool for linguistic research and teaching as well as for practical application.

Data Processing and Analysis

PyPDF2

PyPDF2 is a Python package specifically written for working with PDFs without relying on cumbersome third-party tools. It is able to programmatic extraction of text from PDF, page handling, and other operations on PDFs.

Key capabilities include:

- Text extraction from PDF pages
- Merging multiple PDFs into a single document
- Splitting PDFs into separate files
- Page manipulation (rotation, extraction, merging)
- Adding watermarks and annotations
- Basic PDF encryption and decryption

PyPDF2 is especially useful when working with large documents or batch processing several PDFs, allowing automation of processes that would be time-consuming to do manually.

Pandas

Pandas is an open-source library of software based on Python for data manipulation and analysis, named after "panel data." It offers specialized data structures optimized for tabular datasets with a simple Python API.

The core of pandas is the DataFrame data structure – a two-dimensional, array-like table where:

- Each column represents values of a specific variable
- Each row contains values corresponding to those variables
- Data can include numeric, categorical, and textual types

Pandas enables:

- Importing and exporting data in various formats (CSV, SQL, Excel)
- Data cleaning and transformation
- Subsetting and filtering datasets
- Joining and merging different data sources
- Computing descriptive statistics
- Handling missing data
- Time series analysis and date manipulation

It is efficient in terms of computation for medium-sized datasets, with computationally expensive operations implemented in C or Cython in the backend to optimize performance.

Mathematical and Visualization Tools

Cosine Similarity

Cosine similarity is a mathematical measure of similarity between two non-zero vectors. It is given by the cosine of the angle between vectors, which is the dot product of the vectors divided by the product of their magnitudes.

The resulting similarity ranges from:

- +1: Vectors are exactly the same (0° angle)
- 0: Vectors are orthogonal (90° angle)
- -1: Vectors are exactly opposite (180° angle)

In text analysis application scenarios (e.g., resume to job ad comparison), term frequency vectors never contain negative elements, and therefore cosine similarity may only range from 0 to 1. Thus, this metric is very useful for normalizing document length when comparing, remembering the angle (content similarity) and not magnitude (document length).

Matplotlib & Seaborn

Matplotlib is a rich library for generating static, animated, and interactive plots. It gives explicit control over plot aesthetics through an object-oriented API and is the foundation for many other visualization libraries.

Seaborn is built on top of matplotlib to provide a more high-level interface for statistical plotting. It works well with pandas data structures and provides specialized plots for statistical graphics such as distribution plots, regression plots, and categorical plots.

Together, these libraries allow the creation of sophisticated visualizations to help in the analysis and presentation of resume-job matching patterns and distributions.

Methodology

System Architecture

The code has a modular architecture with clearly defined components such as :

- **Resume Parser**: This reads and parses text from the PDF of the resume.
- **BERT Model**: This performs semantic analysis and similarity tests on the parsed text.
- **TF-IDF Model**: This performs keyword extraction and matching.
- **Resume Scorer**: This uses all the components to finally calculate total scores and provide recommendations.
- **Pipeline Executors**: Provides interfaces for single, batch, and large scale analyses.

Resume Parsing

The resume parser module (resume_parser.py) reads structured data from PDF resumes. Workflow includes:

1. **Text Extraction**: Raw text is pulled from PDF files through the PyPDF2 library
2. **Section Identification**: ResumeParser class recognizes overall resume sections (experience, education, skills, projects, achievements) via regular expressions
3. **Structured Information Extraction**:Precise information is pulled from every section, i.e.,

    a. Education: degree, institution, year
    b. Experience: title, company, duration, description
    c. Skills: list of technical and soft skills

This structured approach allows for targeted analysis of different resume components against specific job requirements.

Semantic Analysis with BERT

BERT model deployment (bert_model.py) uses pre-trained transformer language models to perform semantic processing :

1. **Text Embedding**: Text segment representation in high-dimensional vector space forms.

2. **Semantic Similarity**: Computation of cosine similarity between resume sections and job description sections.
3. **Contextual Understanding**: Inference of experience and skill meaning in context versus keyword matching.

The deployment utilizes the standard "bert-base-uncased" model and is performance-optimized through GPU acceleration where possible. Semantic matching functions allow the system to identify equivalent qualifications even when differing terminology is utilized.

Keyword Matching with TF-IDF

TF-IDF model (tfidf_model.py) for lexical matching and keyword analysis:

1. **Text Preprocessing**: Text normalization and cleaning that removes special characters and stopwords.
2. **Keyword Extraction**: Important words in job postings recognized by TF-IDF score.
3. **Match Scoring**: Resume keywords as a percentage of the job description.
4. **Gap Analysis**: Missing or underrepresented keywords in the resume.

The TF-IDF analysis is further confirmation of the semantic interpretation by finding particular language most likely to be key to pass initial ATS filtering.

Scoring Algorithm

Resume scoring module (resume_scorer.py) aggregates scores from the TF-IDF and BERT models to produce an overall score:

1. **Component Scores**: Calculating individual scores for:
    - Semantic matches (30% weight)
    - Skill relevance (30% weight)
    - Keyword matching (20% weight)
    - Keyword coverage (20% weight)

2. **Overall Score Calculation**: Weighted component scores combined to a single percentage match value.

3. **Improvement Suggestions**: Developing actionable suggestions from gaps found.

Weighting method avoids tipping into favor of one aspect over the other and divides semantic interpretation and keyword availability at equal ratings.

Analysis Pipeline

The software provides some modes of execution:

1. **Single Analysis**: Single resume to one or group of job descriptions
2. **Batch Analysis**: Multiple resumes to group of job descriptions
3. **Extended Analysis**:Executing comprehensive analysis with additional metrics and plots.

Results are saved in multiple formats:

- Summary text files with key findings
- Detailed CSV files with comprehensive metrics
- Visualizations comparing different aspects of the analysis

Analysis of Input and Output

Input Processing

The system takes the following inputs :

1. **Resume Files**: PDF resume files in data/raw/resumes.
2. **Job Descriptions**: CSV files of jobs like title, description, and skills needed.

For example, the sample_jobs.csv stores formatted job descriptions for different jobs like Data Scientist, Software Engineer, and Machine Learning Engineer with full job descriptions and skills needed per job.

Output Generation

For each analysis done, the system will generate:

1. **Summary Report**: A text file containing:
    - Overall match score.
    - Top matching job positions.
    - Key improvement suggestions.
2. **Detailed Results**: A CSV file with comprehensive metrics:
    - Match scores for each job.
    - Keyword coverage percentages.
    - Missing keywords.
    - Improvement suggestions for the resume.
3. **Visualizations**: (For extended analysis)
    - Match score distributions.
    - Semantic score comparisons.
    - Skill relevance charts.

Sample Analysis Flow

Comparing a resume to the Data Scientist job description of the sample dataset:

1. The resume parser gets structured data from the PDF resume.
2. The BERT model computes semantic similarity between job postings and resume sections.
3. The TF-IDF model finds keyword matches and missing keywords.

4. The scorer gives a rate of match measure in general.
5. The system recommends specific items like:
    - Adding missing technical skills (e.g., TensorFlow, PyTorch).
    - Tagging experience descriptions with data visualization
    - Adding measurable results for machine learning projects.

Strengths, Limitations and Future Enhancements

Strengths

1. Dual Analysis Approach: Through the combination of semantic meaning (BERT) and keyword matching (TF-IDF), the system is much superior to traditional ATS systems in analysis.
2. Actionable Suggestions: The system provides suggestions for improvement aside from scoring.
3. Modular Architecture: Since the code is in logical modules, it's easy to expand and upgrade.
4. Support for Batch Processing: The program supports batch processing of many resumes against various job descriptions at once.

Limitations

1. PDF Parsing issues: The system may not support complex resume images or PDF formats.
2. Language Limitations: The implementation is English-optimized.
3. Computational Needs: BERT models are computationally expensive, which could limit how well they work on less powerful computers.

Future Enhancements

1. Improved PDF Parsing: We used improved PDF parsing techniques to support more complex formatting.
2. Extending the system to accommodate resumes and job postings in other languages is known as multi-language compatibility.
3. Developing a web-based interface for easy system use is known as interactive user interface.
4. Resume Generation Assistance: Offering assistance to users to apply the suggested changes.
5. Industry-specific Models: Train industry-specific models to increase accuracy.

Conclusion

Resume ATS Scorer and Matcher offers a top-level solution to the problem of resume optimization for a specific job role. Through the assistance of sophisticated NLP methods, the tool offers greater than simple keyword matching data that informs job applicants about how their resumes are read by contemporary ATS software.

The recommendations which will be presented by the system will allow the users to incorporate specific resume improvement adjustments, which will help them to improve the odds of them getting invited to attend interviews when facing competitive labor markets.

The design and analysis modularity of the features render the system a value tool for career guidance counselors, employment seekers, and hiring consultants. With increasingly more employment being based on automated screening systems, such systems will be necessary in order to enable job applicants to understand the complexities of job searching today.