LLM MID REVIEW 2 Report

Group NoobDe

Methodology for Enhancing Resume

PDF Analysis System for Job Matching and Resume Processing

This code system automates the process of analyzing job-related documents. It takes PDF files as input and performs two main tasks:

1. For Job Descriptions:
- Extracts text from PDFs
- Matches the job to one of 10 predefined roles using semantic similarity
- Uses AI to identify required skills and eligibility criteria

2. For Resumes:
- Extracts and structures personal information
- Breaks down key components like education, skills, work experience
- Organizes projects and additional achievements

All information is processed using AI (via Groq API) and saved in a structured JSON format, making it easy to compare job requirements with candidate profiles.

Resume Enhancement Using Base LLM

This code implements a resume optimization system that tailors a candidate's resume to match specific job requirements. It loads both the job description and resume data from JSON files, then enhances four key sections:

- Skills: Realigns and prioritizes skills based on job requirements
- Work Experience: Highlights relevant experiences and quantifies achievements
- Projects: Emphasizes projects that demonstrate required skills
- Additional Information: Focuses on relevant certifications and achievements

The system uses AI prompts to intelligently modify each section while maintaining the original format, ensuring the enhanced resume better matches the target job position. The optimized resume is then saved as a new JSON file.

Resume Enhancement Using Base LLM and RAG

This code implements a sophisticated resume optimization system using Retrieval-Augmented Generation (RAG). The system works in two main parts:

1. RAG System (First Class):
- Processes PDF documents and breaks them into manageable chunks
- Uses semantic search to find relevant information from a knowledge base
- Maintains conversation history for context
- Leverages LLM (Llama) for intelligent responses

2. Resume Enhancement with RAG:
- Takes both resume and job description JSON files as input
- Queries the RAG system for industry-specific knowledge about the role
- Uses this knowledge to enhance resume sections (skills, experience, projects, extras)
- Makes improvements based on both job requirements and industry best practices
- Saves the enhanced resume with references to sources used from the knowledge base

The key improvement over the previous version is the addition of industry knowledge through RAG, making the enhancements more aligned with industry standards and best practices.

Evaluation

Criteria A: Skill Matching

- Parses and normalizes skills from both resume and job description
- Uses fuzzy matching to account for variations in skill names
- Calculates a skill match score based on required vs. present skills

Criteria B: Grammar

Evaluates writing quality using LanguageTool

Criteria C: Resume Alignment to Job Description

- Keyword Analysis: Extracts and compares relevant keywords.
- Text Similarity: Uses TF-IDF for content comparison
- Semantic Matching: Employs BERT embeddings.

The system combines these factors to generate a final matching score, weighing semantic understanding (40%), keyword matching (30%), and content similarity (30%), providing a comprehensive assessment of how well a resume matches a job description.

Final Criteria

Average of Above 3

Pipeline

Connects all the above components in order and saves the results.
Selects the best model and method as well and saves it.

Results Analysis and Visualization System

This code implements a comprehensive analysis system to evaluate and visualize the performance of different resume enhancement approaches. It consists of:

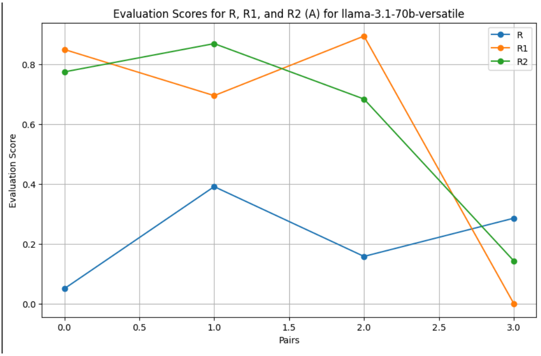
1. Score Analysis:
- Processes three types of evaluation scores (A, B, C) for each version:
  * R: Original resume
  * R1: Basic enhanced resume
  * R2: RAG-enhanced resume
- Calculates average scores across all test pairs

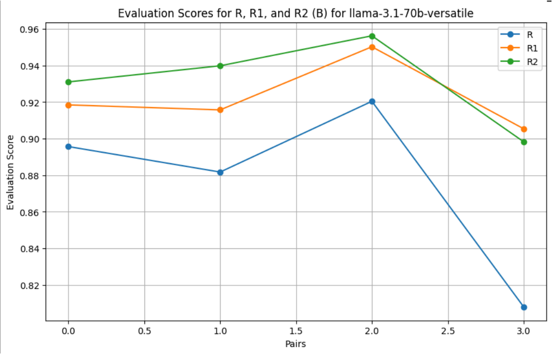
2. Visualization:
- Creates comparative line plots for each evaluation metric
- Shows performance trends across different resume pairs
- Separate visualizations for:
  * Skills matching (A)
  * Grammar quality (B)
  * Overall matching (C)
  * Combined total score

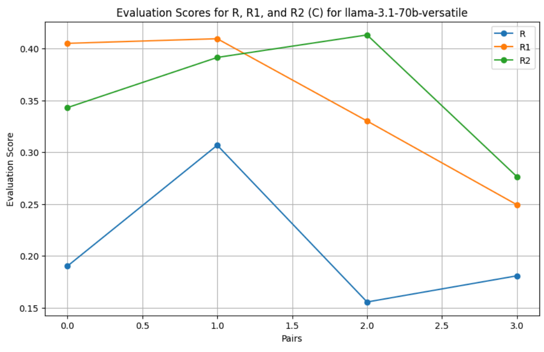
3. Best Model Selection:
- Identifies best-performing combination of:
  * LLM model (70B vs 90B)
  * Resume version (R vs R1 vs R2)
- Stores results for final model selection

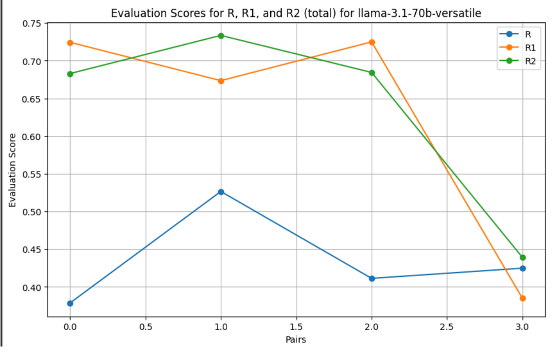
Result-Ouputs of Existing Work

The test set currently only has 4 pairs.

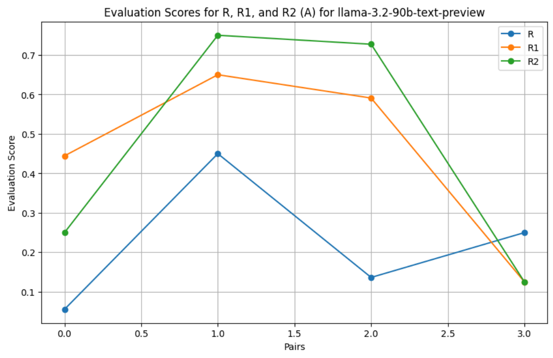


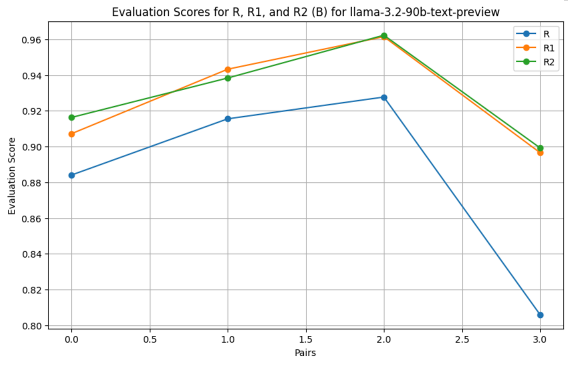


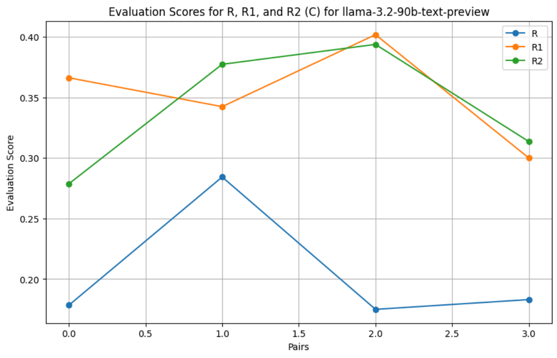


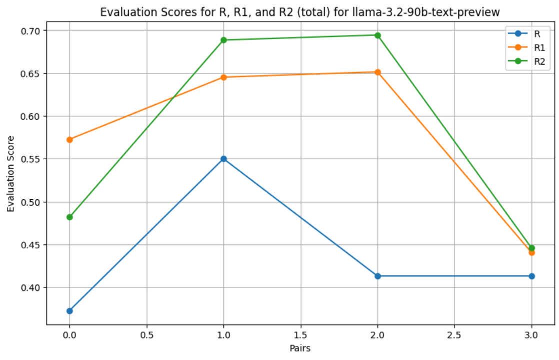


Results for llama-3.1-70b-versatile:
Average Evaluation Score for R: 0.43532647250 93886
Average Evaluation Score for R1: 0.6270193125967944
Average Evaluation Score for R2: 0.6350886409211411









Results for llama-3.2-90b-text-preview:
Average Evaluation Score for R: 0.4371544996145654

Average Evaluation Score for R1: 0.5774336770940104
Average Evaluation Score for R2: 0.5776433689482731

Front-End Application for a User

Hosting

Hosted using Streamlit and Ngrok

Methodology

Users can upload resume pdf and job description pdf
The app selects the best model and enhancement method from the previous step by reading a text file and continues to enhance the uploaded resume.
It outputs an enhanced version of the pdf