Tailored Resume Generation for Job Applications using RAG with LLMs

MSc Research Project
Data Analytics

Komal Prakashchandra Patra
Student ID: x22210369

School of Computing
National College of Ireland

Supervisor: Vikas Tomer

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Komal Prakashchandra Patra
Student ID:	x22210369
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Vikas Tomer
Submission Due Date:	12/08/2024
Project Title:	Tailored Resume Generation for Job Applications using RAG with LLMs
Word Count:	6400
Page Count:	25

Signature:	
Date:	15th September 2024

Tailored Resume Generation for Job Applications using RAG with LLMs

Komal Prakashchandra Patra
x22210369

Abstract

In an era of digital job applications, it is very crucial that the resume should stand out from other applications. The generic traditional resume fails to effectively highlight the skills for diverse job specifications. The research introduces a novel method of tailoring the resume that aligns with job description that leverages Retrieval Augmented Generation (RAG) with Large Language Models (LLM). The resume parsing is the information retrieval step to extract each section such as Education, Experience, skills,etc. which has also been performed using LLMs such as LLAMA-3, Mixtral and Gemma model. For retrieval, the resume data has been stored in Pinecone (as chunked PDF data) and MongoDB (entire JSON resume) vector databases. The Re-ranking techniques such as BM25 and Cross Encoder has been used to enhance the performance of the retriever. In this study, the BM25 which is a keyword-based search outperforms the cross-encoder. THe prompt engineering is used to instruct the model to generate the tailor resume with one-shot approach where one example has been given a model to better understand. The variants of LLAMA-3 were used with ChatGroq to mitigate the computational load and faster inference. To provide the memory to the LLM models, the conversational buffer memory of langchain has been utilised by storing the last five previous chat history, so that the user can keep on refining the tailor resume till they get satified. The efficiency of the methodology is highlighted through a series of evaluations such as RAGAs Metrics, BERTScore, and Custom metrics such as Job Alignment and Content Preservation. By comparing the cosine similarity for content preservation between LLM and RAG with LLM are 72% and 89% respectively. The RAG with LLM outperforms compared to using only LLM to generate the tailored resumes.The Chatbot interface is designed for a user to have a seamless and highly user-friendly experience.

1 Introduction

Today, being in this extremely competitive job market, securing a job has been extraordinarily difficult. Crafting a resume that is eye-catching and customized according to the different job specifications is the very first and most important step; now, since the generic resume no longer holds its value as the high overview of a candidate's qualification often fails to bring into light some particular skills aligning with job specifications. The traditional method involves manual tailoring which leads to human error as well as its time-consuming. There can be several reasons such as not knowing the trends and patterns as well as the best practices of writing a resume which leads the job seekers to rejection. The motivation behind this research was implementing AI strategies to overcome the challenges and assist job seekers in tailoring resumes for multiple job applications, which triggered the need for an efficient, automated solution to enhance the relevance and quality of resumes, thus improving job search success.

Recent developments after Generative AI and Natural Language Processing (NLP) come into picture which opens several possibilities for automating various applications such as resume scoring, identifying key matches, resume evaluation based on job description (JD) and job recommendation based on resume. Despite these developments, there has been a lack of easy-to-use, convenient tools that would just bring all these capabilities together for job seekers. The benefits of this research are both academically and industrially. Academically, the project benefits the growing research in AI application as well as the career development and job search optimisation. Industrially, there will be a potential increase for job seekers getting an interview and a job while the employers receive a more relevant and well-tailored resume.

While some research has attempted to generate the resumes directly through the large language models (LLMs) similar to the previous research done by the Author (Zinjad et al., 2024), the existing solutions often lead to lack of domain specific knowledge and can cause hallucination. Additionally, the authors have used 2 state-of-the-art models: Gemini-pro and GPT which makes them inaccessible. So, in order to fulfill these gaps, the research has addressed a more effective approach that has incorporated domain-specific source knowledge such as Retrieval Augmented Generation (RAG) gained attention in late 2020s (Lewis et al., 2020) and by utilizing open-source state-of-the-art LLM models.

1.1 Research Question

How best a RAG can effectively integrate with open-source state-of-the-art LLM models to enhance the accuracy and relevance of tailored resume generation with considering the end-user feedback to be effectively taken onboard into conversational memory to refine the tailored resume?

1.2 Research Objectives

In order to further fulfill and implement the objectives of the research questions, the below strategies have been outlined to achieve and establish the study:

1. Perform resume parsing using various LLM models to assess and compare their performance in terms of extraction accuracy.
2. Implement RAG and develop a system to store all the parsed resumes into a document structure format with their embeddings for efficient retrieval.
3. Apply one-shot prompt technique so that the retrieved response from RAG can guide the LLM model to generate the tailor resume.
4. Evaluating the embedding-based and RAGAS metrics based on 2 re-ranking strategies with diffrenet chunk sizes.

5. Design a conversational Buffer memory interface through which the user can modify their resume until they get satified with the generated resume. To experiment, this may require iterative prompts.

6. Define performance metrics to evaluate both the retrieval and Generation Process.

7. Design a user-friendly UI interface for the chatbot using Streamlit. Make the chatbot conversational so that user can refine the modification.

1.3 Structure of the report

The report move to a related work that has been performed to personalised the resumes using LLMs by previous authors. Further, the methodology design helps us to understand the core component of the research from data collection to Tailor resume generation. In order to understand the flow and blueprint of a project, the Design Specification focus on components like Resume Parsin, RAG (Retrieval and Generation process), Conversational Chain memory and the evaluation using Performance Metrics. All this leads us to the result section, where the findings and experimentation has been focused on with the implemntation of chatbot. Lastly, The reserach isconcluded with all the discussion and finding with the the limitation for the scope of Future work.

2 Related Work

The Paper expressed the potential need of tailoring the resume as per job specifications to fit the job profile. The (Zinjad et al., 2024) has used LLM models such as OpenAI GPT-4 and Google's Gemini which leads to a cost-intensive approach from the users perspective. The ResumeFlow application consists of the User extractor which converts the resume PDF into JSON format by extracting the essential information from the resume. The ResumeFlow application consists of the User extractor which converts the resume PDF into JSON format by extracting the essential information from the resume.

These papers discuss the limitations such as the model that are not open-source, including the potential issues with the maximum context length as well as the LLM model that leads to major concerns of hallucination. Overall, The ResumeFlow presents the possible way of solving the typical problem that job seekers encounter, with the potentiality of using LLM.

This paper (Sunico et al., 2023) has focused on assessing the resume along with the generation using LLM. Models such as OpenAI GPT-3.5 , Llama and BARD have been utilised for generating the resume, For Resume generation, the application takes each section of the resume and provide various possibilities of bullet point that suits with job description. The Author has used prompt approach which are effective and crucial for LLM. Multi-prompt methods combines 2-3 different categories that are newly developed category of prompt that utilises the several prompts such as prompt ensembles and prompt composition in order to increase the performance of the model. Again, directly using LLM sometimes giving irrelevant output ussualy known as "hallucinated" outputs that misinterpret the skills and experiences from the resumes.

The author presents (Bruera et al., 2022) the methodology for generating the "synthetic resume" while ensuring the strong ethical consideration through differential privacy (DP) and GDPR regulations. Firstly, this research has extracted the candidates attributes from real CVs by downstreaming the NLP techniques such as NER, relation extraction and part-of-speech tagging. Secondly, the Bayesian network has been used to model the conditional dependencies between the candidate attributes. Lastly, the resume generation process using the GPT-2 model with the specialized prompt techniques ensures resulting CVs are both realistic and privacy-preserving. In case of evaluation, the author has used fastText (CBOW) and BERT model based on generated and the Augmented resumes. The limitations are the generated resumes are not totally generated as per job specification and only relies on normal word embeddings.

This paper focuses on developing the chatbot user interface for Generating the resume using the LLM and the Latex technologies for converting the generated resume output into PDF LATEX templates. The authors have used the LSTM model specifically involving the series of operations to enhance conversational capabilities of chatbot. The datasets being utilized in this research are the transcripts of human-to-human conversations and chat logs from past interaction. The LSTM model has undergone several epochs of training and several one-shot prompt approach to lead the response process generated by the chatbot. However, it is important to recognize and evaluate the limitations and biases. Human oversight element is also important to control and confirming the correctness and relevancy of the generated resumes.

The Author (Balaji et al., 2019) for the paper AIResume automates the generation and refinement of work history sections for resumes using Natural Language Generation (NLG) and Information Exxtraction (IE) technologies. The data was collected by leveraging the CareerBuilder database from which AIR provides personalized suggestions for job titles and unstructured data through clustering, advanced resume and job parsing. The AIR has use the POS (Part-of-speech) tagging in order to tailor the work history. AIR's advantage is that it aids in the easier preparation of resumes, with an emphasis on mobile users. Thanks to the AI writing assistant okay, there has never been so effective a text creation tool designed for this purpose. It is this novelty that enables it to set out the earlier attempts at doing the same thing of composing the work of the player with the help of automation.

The paper (Kale and Andreopoulos, 2023) proposed the methodology revolving around the advanced NLP techniques to generate the job-tailored resume content. The 1000 of resume dataset was gathered from the students through google form while the job description data were web-scraped. The data which include job descriptions and its corresponding resume content was fine-tuned using the transformer models like BERT and GPT-2. As a result, implementing the fine-tuning methodology was able to generate a resume that was coherent and contextually relevant content that is more aligned with job descriptions compared to the generic resumes. The major limitation was truthfulness of the generated resume as it should not exaggerate the candidate's qualification which can lead to ethical concerns. Therefore evaluating the application with some performance metrics can help to enhance the factual accuracy in the generated content.

The study explores the effectiveness of using RAG with LLMs to reduce the hallucinations. The (Foulds et al., 2024) used an experimental method where human participants performed the evaluation of accuracy for the generated response by an LLM under two critical conditions: with and without context. The study collected 336 pairs of prompt-responses where the accuracy was calculated based on how well the model used the context provided to generate correct or hallucinated responses. They have used the GPT-3.5 Turbo model along with RAG. The limitation was human evaluations as it introduces subjectivity and variability which can be solved by exploring the RAG in more diverse contexts with its impact.

Author & Year	**Functionality**	**Models Used**	**Limitations**	**Solution Proposed**
Zinjad et al. (2024)	Tailored resume generation from the job descriptions	OpenAI GPT-4 model and Google Gemini's Pro	Hallucinations & Inaccessibility	Use of retrieval augmented generation (RAG) and Open-Source Models
Sunico et al. (2023)	Resume Generation, Resume Assessment, and chatbot user I/O	GPT-3.5 Turbo	Hallucinations & Inaccessibility	Use of RAG, evaluations, and Unstructured data
Sumedh et al. (2023)	Tailored resume generation from the job descriptions	GPT-2	Complexity issue related to memory, No Evaluations	Evaluations of LLM and AI inference tool
Naveena et al. (2023)	Tailored resume generation from the job descriptions	CNN and BERT	Only basic word embeddings	Use of RAG or fine-tuning
Ram Kishor et al. (2024)	Tailored resume generation from the job descriptions	LSTM	No validation of generated resume	Evaluation metrics
Janani et al. (2019)	Generating work history section for resume	NLP (POS tagging)	No advanced resume parsing, and only focus on work history	Using LLM for resume parsing
Philip et al. (2024)	Study on LLM and RAG on resume/CV data	GPT-3.5 Turbo and RAG	Hallucinations, noisy context, mismatched instructions and context, context-based synthesis	Use RAG to avoid the hallucinations

Author & Year	**Functionality**	**Models Used**	**Limitations**	**Solution Proposed**
Skondras et al. (2023)	Resume classification	BERT and FFNN	They are only generating resume dataset	Use different variant of BERT that aligns with specific challenge called JobBERT
Chengguang et al. (2024)	Resume screening/assessment	LLaMA, BERT, and T5	Generated datasets were not annotated	Datasets can be annotated and the newer variants of BERT can be applied for better results
Vaishali et al. (2023)	Enhancing job recommendation through resumes	GPT-4	Implication of wrong recommendations	Fine-tune the prompts incorporating few-shot problems and leverage reinforcement learning
Shiqiang et al. (2015)	Enhancing job recommendation through resumes	Semantic Similarity Search	No generative AI has been used, purely developed on traditional NLP approaches	Use LLM and RAG
Alessandro et al. (2021)	Summarizing the resumes	BERT and hierarchical clustering	Have only performed BERT-based embeddings	Fine-tuning the BERT model
Pradeep et al. (2020)	Recommendation system	NLP, text classification, and ML models, and content-based recommender model	They have used CSV data while they could have used PDF or Word documents	It can be enhanced by using RNN or LSTM
Yu-Chi Chung et al. (2022)	Resume classification	Bi-LSTM, Graph Convolution Network (GCN), and dotGAT	Using DL model	Downstream tasks can be performed such as tailoring the resume

3 Methodology

This project uses the combination of RAG with LLM to develop a system that customises the resume based on the job specifications. The methodology depicted in a Figure 1 is structured in a way that the resume is not only relevant but can also be tailored and optimised on the diverse specification of job application. Moreover, the user can also can keep on refining the Tailored resume considering the previous changes reflecting using the memory for an LLM.

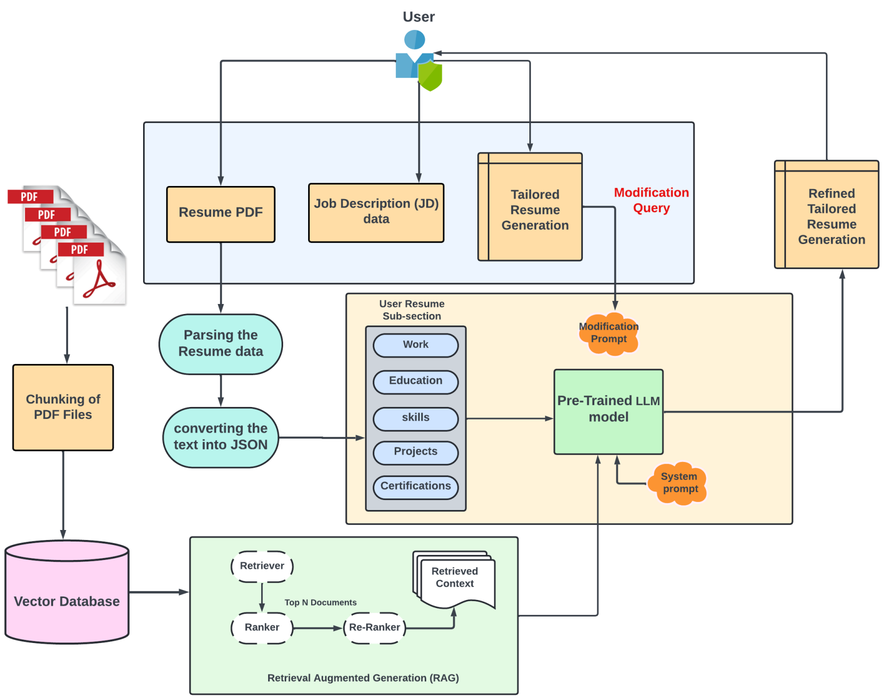


Figure 1: Research Architecture for Tailored Resume Generation

3.1 Data Collection

For the study of Tailored resume generation, the Resume Dataset has been utilised which is available at Kaggle [1]. The dataset contains a variety of resumes representing different domains such as HR, Advocate, Engineering, IT, etc, Each of the Resumes consists of sections like Professional Summary, Education, Experiences, Skills and Achievements. The data contains approximately 2500 resume PDFs that do not contain any confidential personal details such as name, email address and contact information.

3.2 Data Preprocessing

Any NLP task that involves a long document may require pre-processing or transformation in order to improve the accuracy or efficiency of the task [2]. As the data consist of sub-folders representing the domains inside which the PDF files are stored. To load

[1] `https://www.kaggle.com/datasets/snehaanbhawal/resume-dataset`
[2] `https://medium.com/@sushmithabhanu24/retrieval-in-langchain-part-2-text-splitters-2d8c9d595cc9`

entire Directory, the `DirectoryLoader` from `langchain` has been used to load the directory. Recursive Text Splitter can be effectively used when there is a need to analyze the large document which is further divided into chunks for an analysis, which makes more precise and comprehensive breakdown possible compared to standard granular level analysis. Recursive Text Splitter divides the text into pieces by using separators [“\n\n”, “\n”, “ ”, “] in an iterative way addressing the text that is longer than the chunk length without a separator. The chunk size and the chunk overlap are critical hyperparameters that need to be carefully considered as it can directly impact the performance of the generated responses. The chunk overlap is the number of characters overlapping between the two chunks. Thus, no necessary contextual information is lost between chunks.

3.3 Embedding Models

Embeddings are the numerical representation of converting the word or sentences into dense vectors of a fixed dimensionality that include their semantic meaning and the relationship within a specific given context. HuggingFaceBgeEmbeddings is a class with the Langchain community module that allows the usage of models from the Hugging face repository to generate the embeddings. The model processes the input text from the chunk and uses its learned parameter to convert each sentence into a dense vector representation.

3.4 Retrieval Augmented Generation (RAG)

The Most popular AI framework called Retrieval Augmented Generation (RAG) improves the quality of LLM-generated responses by providing the external source of knowledge to a specific domain to narrow down and provide the relevant contexts to the LLM model [3]. The RAG consists of two sub-process as follows: (i) Retriever which initially takes the user query (input) and searches the top-k relevant data; (ii) then, the original user query and the most top relevant retrieved documents are fed into the Generator which is an LLM model responsible to produce the desired generated output (Zhao et al., 2024). Sometime directly using Transformer-based LLM model leads to hallucinations because most of these models are trained on a broad range of openly available public data due to which the LLM cannot access data beyond their training data causing them to respond inaccurately, giving the outdated response or hallucinate and start making their answer which is factually not true. So making use of RAG benefits by providing up-to-date and precise responses, reducing hallucinations and providing domain-particular responses [4]. The RAG consists of two types of Retrieval metrics; Sparse-vector retrieval and Dense-vector retrieval. Retrieval of the sparse vector, for instance, TF-IDF and BM25 is a usual way to compare similar items, since they efficiently consider the keywords using the inverted index. Additionally, the variants of dense-vector retrieval methods have the same fundamental grounds and consider the contextual parameters by using sentence embedding vectors. The texts are converted into low-dimensional vectors using BERT-based encoders and Cross-encoders. The retrieval scores are computed using inner products among the vectors (Li et al., 2022).

[3]https://research.ibm.com/blog/retrieval-augmented-generation-RAG
[4]https://www.databricks.com/glossary/retrieval-augmented-generation-rag

Querying the input vectors helps to identify the relevant matches based on similarity metrics such as cosine similarity and dot product. The retrieved documents are the top most similar documents but are not in ranked format which can wrongly influence the generated responses. To resolve this issue, A Reranker is a technique that reassesses and re-ranks the documents retrieved from the initial search with respect to the query. The Rerankers can be utilised to bring significant improvement in the precision of the retrieved information, which in return enhances the overall performance of the given RAG system [5]. Hybrid search is the kind of RAG search technique which combines two or more searching algorithms to enhance the relevance of search results. Hybrid search is the most common combination of traditional keyboard-based searches and vector search. The keyword-based search considers the capability of exact keyword matching while the semantic vector search performs the search not only in language but also in multilingual and multimodal. The combination of a search executed both through vectors and using keywords enables us to deliver the job description with a semantic match to the users from the resumes stored in Vector DB and at the same time, traditional keyword search is used by the users who prefer it and are already familiar with it [6].

3.5 Model Training

The Large Language Model (LLM) is intended to interpret, generate, and manipulate language-related tasks at the highest level. Self-attention mechanisms are a type of transformer-based architecture that is preferred by most of the language models such as GPT, LLAMA, Gemma, Mistral, etc, to process text and generate text. LLMs can generate human-like text in response to a given instruction prompt. There are models which have encoder-decoder architecture and some of them consist only one of them in their architecture. These models are helpful to develop powerful chatbots and virtual assistants, enabling them to understand context-aware conversations with users.

LLAMA-3 Model: The Large language model Meta AI (LLAMA) is a state-of-the-art model developed by Meta AI. There already exist previous models such as LLAMA and LLAMA2 with billions of parameters that try to understand the contextual meaning. Recently Meta released the LLAMA-3 model, pre-trained and instruction fine-tuning language model variants with 8B and 70B parameters [7]. The LLAMA model has 2 variants, the base model is pre-trained on trillions of publicly available open source data while the Instruct model has been fine-tuned from the base model for helpfulness and harmfulness, utilising the combination of supervised fine-tuning and reinforcement learning. The LLAMA 3 model uses tiktoken as the tokenizer with the context windows of 8192.

3.6 Prompt Engineering

One can craft the basic prompt that enables LLMs to produce high-quality responses by giving clear and precise instruction which involves formulating prompts that are unambiguous and specific which guides the transformer-based LLM model towards generating the desired output. In Prompt engineering, there are various types of prompt techniques

[5]`https://www.webuters.com/beyond-basic-rag-leveraging-rerankers-and-two-stage-retrieval-for-deepe`
[6]`https://www.webuters.com/beyond-basic-rag-leveraging-rerankers-and-two-stage-retrieval-for-deepe`
[7]`https://ai.meta.com/blog/meta-llama-3/`

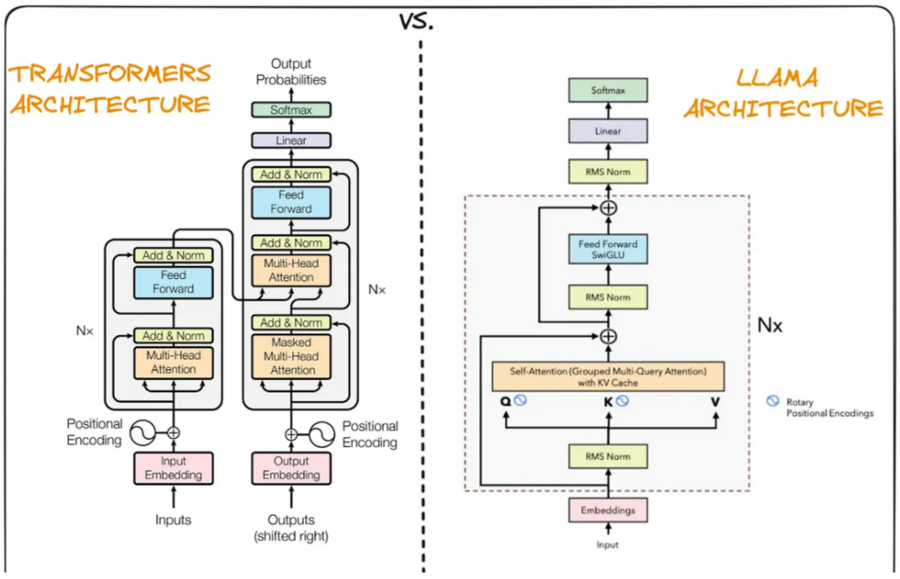


Figure 2: LLAMA Architecture

such as one-shot and few-shot. For a one-shot approach, one example is given as a reference to a model to understand the input and how to move further. While in a few shot approaches, multiple examples are provided. This study utilises the one-shot approach where the templates include the instruction for what to do, specific context coming from the RAG model and questions as the job description with some one-shot examples.

3.7 Model Evaluation

The performance of LLM is enhanced by RAG by downstream the domain-specific tasks, providing the external source knowledge base. The RAG system consists of two components; Retrieval and Generation. Evaluating RAG systems involves evaluating the performance of RAG systems either by retrieval, generation or in general, the RAG system as a whole (Yu et al., 2024). The LLMs has the ability to generate the text based on extensive information but sometimes the generated text hallucinates and produces incorrect, misleading or biased information. Evaluating the RAG systems is very essential so that the retrieved document from the vector database remains relevant and the generated text is accurate, by effectively combining the retrieval and the generations. The vastness requires evaluation metrics that can effectively measure the precision, recall, F1 score and relevance to evaluate the retrieved documents in the context of the job description as a user query. There are various framework/tool which evaluate the performance metrics such as RAGAS, BERTScore, DeepEval, ChainPool, and EvidentlyAI.

The aim is to ensure that your RAG system answers correctly, which could be both relevant and challenging to a dataset. The several methods to evaluate are embedding-based evaluation, RAGAS evaluation and LLM as a judge. For embedding-based evaluation, the method uses model embeddings to evaluate the precision, recall and relevance score for the responses from the RAG systems, but the only limitation is that it might miss the delicate nuances of complex queries. While the RAGAs is a framework that helps to evaluate the RAG pipelines which aggregates the scores for faithfulness, relevance, context recall, answer correctness, and context precision. Using one LLM to evaluate the performance of another implies prompt engineering to provide detailed feedback, highlighting the correct and incorrect responses as well as the hallucination.

4 Design Specification

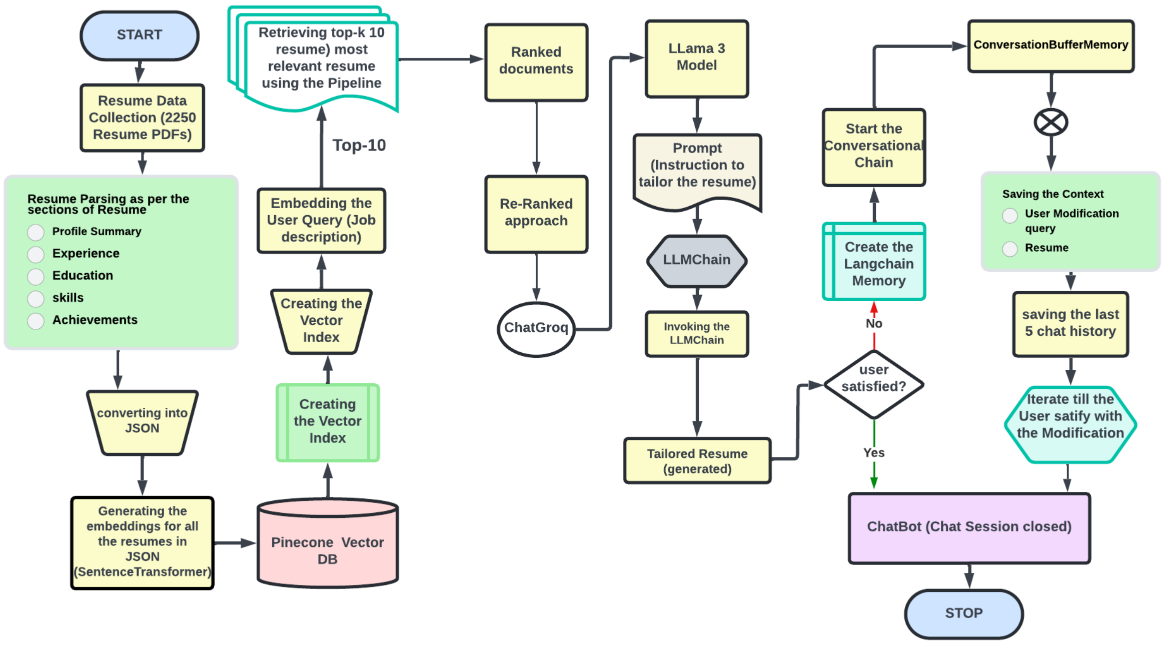

START
Resume Data Collection (2250 Resume PDFs)
Resume Parsing as per the sections of Resume
Profile Summary
Experience
Education
skills
Achievements
converting into JSON
Generating the embeddings for all the resumes in JSON (SentenceTransformer)
Pinecone Vector DB
Creating the Vector Index
Creating the Vector Index
Embedding the User Query (Job description)
Top-10
Retrieving top-k 10 resume) most relevant resume using the Pipeline
Ranked documents
Re-Ranked approach
ChatGroq
LLama 3 Model
Prompt (Instruction to tailor the resume)
LLMChain
Invoking the LLMChain
Tailored Resume (generated)
user satisfied?
No
Yes
Create the Langchain Memory
Start the Conversational Chain
ConversationBufferMemory
Saving the Context
User Modification query
Resume
saving the last 5 chat history
Iterate till the User satify with the Modification
ChatBot (Chat Session closed)
STOP

Figure 3: System Flow Design

4.1 Resume Parsing

The resume taken from the user as input is further abstracted as raw text using the PyPDF2 library. This processing at the textual level is a guarantee that the resume is properly extracted from all the resume sections. The low level design for the detailed understanding of resume parsing process can be seen in Figure 4. First, when the text is extracted, the second step is to enrich and format it through a predefined resume template. This is where the Pydantic comes into the picture, It is a free dependency library for data validation and setting management tools. It helps in establishing and defining the data model that matches the expected format of resume data because it ensures that all required fields are appropriately filled out and correspond to the expected structure. The system allows an option to choose a range of LLM models for parsing and converting the resume data. The models are Groq's less computationally intensive approach, namely Gemma 7B, Mixtral 8*7B, Llama 3 8B, and Llama 3 70B. Each model has its capabilities with pros and cons, which affect their resumes. AI inference from Groq gives a fast performance. The Groq LPU™ AI Inference Technology achieves outstanding computer speed, quality, and energy efficiency [8]. The Benefits of using Groq's are scalability, efficiency, contextual parsing, enhanced Accuracy, cost-effectiveness, reduced computational

[8]https://rito.hashnode.dev/building-rag-in-2024-with-langchain-groq-llama3-and-qdrant

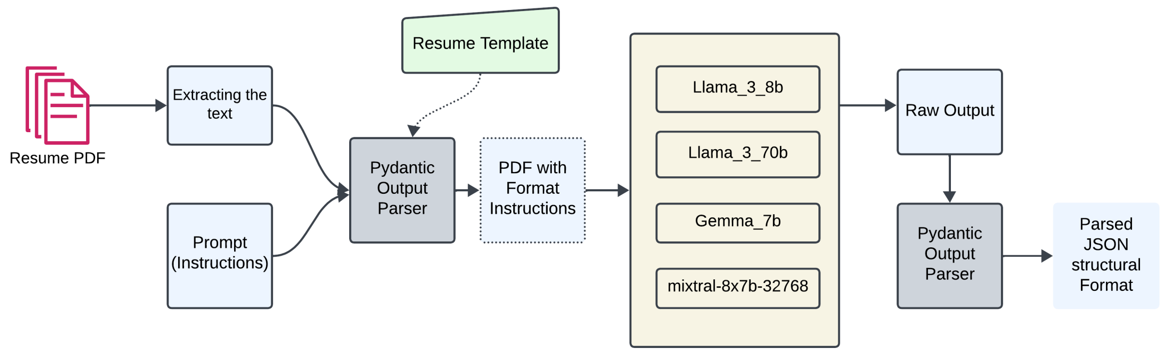


Figure 4: Design Flow of Resume Parsing component

load and compatibility. Upon the selection of an appropriate model, the resume data gets analyzed to get helpful information which is given in a well-structured JSON format.

4.2 RAG

4.2.1 Retrieval Process

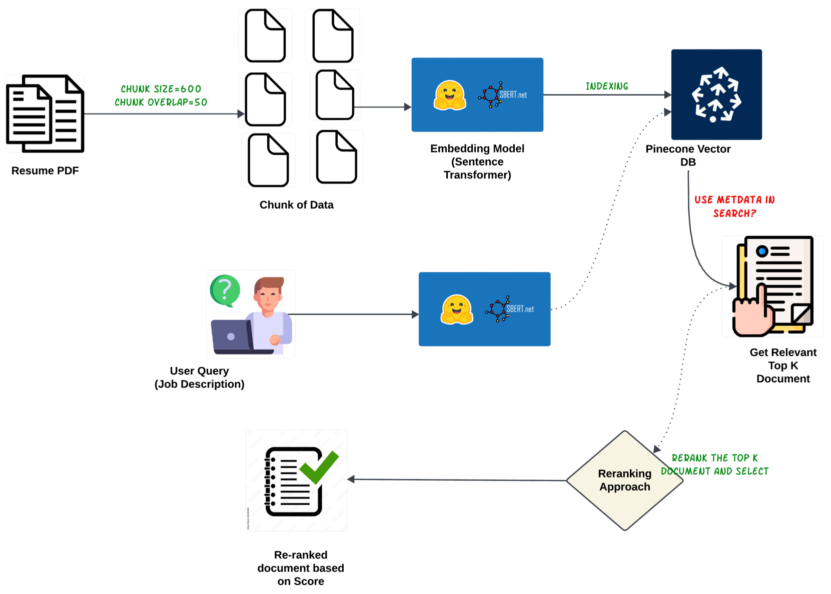

Resume PDF
CHUNK SIZE=600
CHUNK OVERLAP=50
Chunk of Data
SBERT.net
Embedding Model (Sentence Transformer)
INDEXING
Pinecone Vector DB
USE METDATA IN SEARCH?
Get Relevant Top K Document
User Query (Job Description)
SBERT.net
RERANK THE TOP K DOCUMENT AND SELECT
Reranking Approach
Re-ranked document based on Score

Figure 5: Design Flow for RAG component

The chunks of data received after performing the data pre-processing step are then used for the embedding to convert those chunks into dense vectors with a dimensionality size of 1024. The embeddings are stored in the Pinecone Database by creating the Index. Pinecone is a vector database designed for storing, indexing and querying large scale vector data for high-performance similarity searches. The Pinecone index configuration consists of the dimensionality of the vector as 1024, as the embedding was created using the hugging face embedding model was similar. The distance Metrics for similarity search is “cosine” which evaluates the correlation between the directional vectors by cosine similarity. The MongoDB vector database is considered to store the entire resume as a one document where the vector pipeline consist of 130 number of candidates with top 4 resumes to retrieved.

RAG is a hybrid strategy combining Information retrieval and Natural language generation to enhance the understanding of AI language models. RAG consists of two-step processes: Retrieval and Generation. The retrieval steps help the generative model to fetch the top K relevant document from an external knowledge base so that the generative model can access specific and detailed information beyond its training data. The Job description acts as a user query for the retrieval system, further converted to vector embeddings and then transferred to Pinecone Vector DB. Relevance is more frequently used to determine the true values in metrics such as cosine similarity or other distance measures. The top 10 resumes are retrieved from Pinecone as the most similar documents matching the job description. They are first fetched and then assessed and assigned scores depending on their cosine similarity. Cosine similarity is a measure of how similar two vectors are to each other by calculating the cosine of the angle between the two vectors.The formula for cosine similarity score is as below:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$ [9]

Based on the similarity scores, the top 10 documents are ranked from top to bottom, so that the generative model can understand which document contains more context about the user query.

i) BM25: BM25 is a probabilistic retrieval model that ranks the document on the basis of frequency and relevance to query terms they contain. Each relevant document that is retrieved in the candidate set is scored using BM25 formula. The BM25 score for a document d with respect to a query q is calculated as:

$$BM25 = \sum_{i}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{\text{field\_len}}{\text{avg\_field\_len}}\right)}$$ [10]

The BM25 score represents the document's relevance to the query based on term frequency, inverse document frequency, and document length. The top 10 retrieved documents are then reordered descendingly based on their BM25 scores. Higher scores are indicating higher relevance.

ii) Cross Encoder: The input for Cross Encoder architecture mode consists of the data pairs (2 sentences or 2 documents). They describe the relationship and interactions between the two input sequences. The encoder is made up of a number of neural network units and RNN (Recurrent Neural Network) layers that encode information from unit sequences into fixed-size representation. In this process, each of the retrieved top resume documents is paired with a user query (i.e. job description) and are feeded into a cross encoder model. The model understands the context and relevance for each of the document pairs after which the model provides the relevancy score. These scores are

[9] https://towardsdatascience.com/cosine-similarity-how-does-it-measure-the-similarity-maths-behind

[10] https://www.elastic.co/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables

reordered from high to low which are then transferred to the generative model.

iii) Hybrid Search with Cohere API re-ranking: The semantic search often leads to a lack of inefficiency and precision. Hybrid search, which is a mix of the different search methodologies, enhances the productivity and precision of RAG systems working on several data repository problems[11]. In the basic RAG system, the embeddings are used for the semantic search over the keyword search, but both of them have their own pros and cons. The top K retrieved documents are then given to the Cohere API approach for re-ranking. The ContextualCompressionRetriever is provided with an ensemble retriever and base compressor to get the relevancy score and is given as input to the generative model.

4.2.2 Generation Process

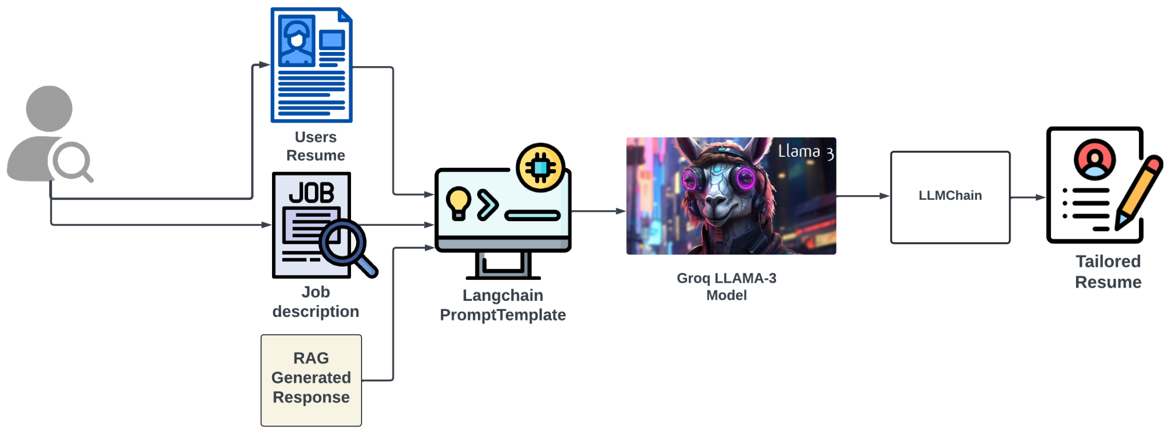


Figure 6: Design flow for RAG generation process

After the Retrieval process, the resume taken from the user along with the job description as input is used by the LLAMA model. All these inputs are not directly provided to a model, instead, the PromptTemplate helps to create the instruction and structures the input in a way that the Model can understand better and tailor the resume in a proper format to satisfy every condition mentioned in a prompt template. By providing precise instruction through prompt templates, the models give attention to essential information and generate outputs according to the needed structure by giving a well-organized layout. It specifies how the job description and the candidate's resume details should be fed into the model to produce the tailored resume. The Groq is integrated with Langchain by using Groq cloud, models like LLAMA, Gemma and Mistral can be called through API which helps us to computationally run highly computed models with billions of parameters in a very short span of inference time without any cost getting involved. LLMChain consists of sequences of components that need to be executed one after another to generate the desired output with the given structure[12]. The essential component, the Prompt template is required to structure the input in proper order, the retrieved document from the candidate's set as well as the new input is directly taken from the users which is their

[11]`https://superlinked.com/vectorhub/articles/optimizing-rag-with-hybrid-search-reranking`
[12]`https://medium.com/@minh.hoque/what-are-llm-chains-671b84103ba9`

resume that they need to tailor. The LLM chain further performs some additional preprocessing and formatting to give all these inputs to the ChatGroq LLAMA model. This model can understand the instructions and the contextual relevance from the inputs and generates the modified tailor resume. The LLM chain has different types of chain; Stuff, Map-reduce and refine. For this study, we have used a stuff chain because it handles the large documents which are divided into smaller chunks and then used for the semantic search techniques to retrieve the relevant documents from it to generate the response. However the limitation behind using this stuff chain is its complexity and the potential loss of contextual coherency.

4.3 Conversational Memory

Understanding and utilising the feature of Conversational memory helps to develop the interactive and smart chatbot. These memories have the ability to remember the past interaction that happened between the AI and the human users, which is paramount. Langchain memory stores the conversational history for the model to understand and get the context of previous chats to understand and refine the generated responses based on it. The Conversation buffer memory is a very simple form of memory which stores and maintains the list of conversations in the form of a buffer and passes them to the prompt. Every time one human interacts with an AI, the conversation is recorded in the history parameter. The main purpose behind it is to refine the tailored resume as per the modification needed by the user until they are satisfied.

4.4 Evaluation Metrics

This study focuses on assessing three different types of metrics to understand the performance of the Application.

4.4.1 Embedding-based Metrics

The method uses contextual embeddings and thus handles a set of reference sentences and candidate sentences to calculate the cosine similarity (Hanna and Bojar, 2021). To compute the BERT score, we first need to feed the reference (x) and the candidate ($\hat{x}$) into a BERT model, where the reference represents the top K retrieved documents and the candidate represents the response from the RAG system after generation. Retrieval Precision is the percentage score of retrieved context that is relevant to the response of the relevant user query. Retrieval Recall, on the other hand, measures the reference tokens captured by the generated tokens that have semantically similar tokens in the candidate token.

$$\mathrm{RBERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i \cdot \hat{x}_j \qquad \mathrm{PBERT} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i \cdot \hat{x}_j$$

[13]

$$\mathrm{FBERT} = \frac{2 \cdot \mathrm{PBERT} \cdot \mathrm{RBERT}}{\mathrm{PBERT} + \mathrm{RBERT}}$$

The F1 score is a metric that is measured using the harmonic mean of precision and recall along two axes, thereby reflecting the accuracy (precision) and completeness (recall)

[13]https://aclanthology.org/2021.wmt-1.59/

of the generated text. However, sometimes this metric can be misleading if we allow for the possibility that the text generates covers more of the reference than any other text it generates.

4.4.2 RAGAs Metrics

There are several metrics that are necessary for evaluation, therefore RAGAS [14] has the ability of assessing different dimensions without relying on human-provided ground truth annotation.

i) Faithfulness

It measures the factual accuracy and consistency of the generated answers against the given context. These estimations are very essential, because of the fear of visual hallucinations.

$$\text{Faithfulness} = \frac{\text{Number of verifiable claims in generated response from given context}}{\text{Total number of claims in generated response}}{}_{15}$$

ii) Context Precision:

Context precision assesses whether all ground truth relevant items present in the retrieved contexts are ranked higher. Ideally, all the relevant chunks must appear at the top rank and use the user query and the retrieved context from the vector database to calculate the signal-to-noise ratio.

$$\text{Precision@K} = \frac{\sum_{k=1}^{K} (\text{Precision@k} \cdot v_k)}{\text{Total number of relevant items in the top K results}}{}_{16}$$

iii) Context Recall:

It measures the degree to which the retrieved context matches the annotated answer, known as the ground truth. Context recall is a RAGAS-only metric that requires ground truth, which limits its applicability as annotated ground truth data may not be available for every application.

4.4.3 N-gram based metrics

i) BLEU

The BLEU [17] score analyzes and evaluates the text quality mainly by checking the n-grams in the candidate text and comparing them to the n-grams in the reference text. In the BLEU system, the ideal score range is from 0 to 1, where 0 indicates no reference match, and 1 indicates an exact match.

[14]https://docs.ragas.io/en/stable/concepts/metrics/

[15]https://docs.ragas.io/en/stable/concepts/metrics/faithfulness.html

[16]https://docs.ragas.io/en/stable/concepts/metrics/context_precision.html

[17]https://docs.kolena.com/metrics/bleu/

ii) ROUGE

ROUGE [18] is widely used for evaluating the quality of natural language text by looking at n-grams, word sequences, and even the longest common subsequence in both the candidate and reference texts.

4.4.4 Custom Metrics

i) Job Alignment

The Job Alignment is a measure that estimates the proximity of the resume and job specifications by assessing both the token and latent spaces similarity.

$$\text{job\_alignment\_token} = \frac{|W(D_{\text{gen}}) \cap W(J)|}{\min(|W(D_{\text{gen}})|, |W(J)|)}^{19}$$

where D_{gen} is the tailored resume, J consists of the job description, and $W(x)$ refers to the number of unique words in x.

ii) Content Preservation

The Content Preservation is meant to evaluate the system's ability to preserve the content of the user's own resume in the generated one so as to identify and solve any arising problems.

$$\text{content\_preservation\_token} = \frac{|W(D_{\text{gen}}) \cap W(D_{\text{user}})|}{\min(|W(D_{\text{gen}})|, |W(D_{\text{user}})|)}^{20}$$

where D_{gen} consist of tailored resume while D_{user} consist of the resume provided by the user initially and $W(x)$ refers to number of unique words in x.

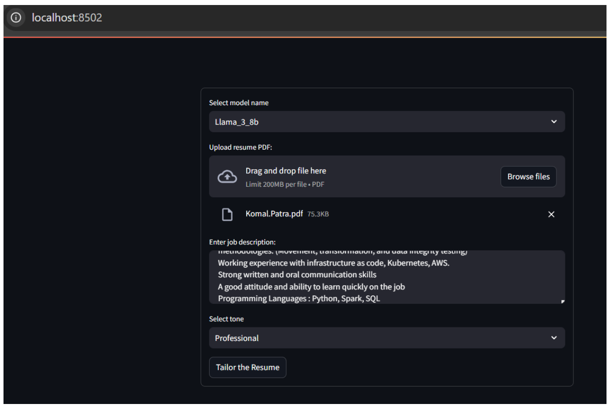
5 Implementation

The Chatbot was developed to optimize the job application by tailoring the resumes to job specifications, as illustrated in Figure 7. The chatbot allows users to upload their existing resumes and job descriptions. The chatbot parses the resume for extracting the information for each sections. The parsed data are structured in a JSON format which is further processed by LLM model using which the model tailor the resume accordind to job specifications. Additionally users can further refine their resumes by directly providing the input with the modifications needed and improving the resumes to increase the chance of getting an interview.

The generated resumes come as a JSON output which in future can be converted into several resume LaTeX templates. Below is the generated tailored CVs as per one of the job description:

[18] https://docs.kolena.com/metrics/rouge-n/

[20] https://arxiv.org/abs/2402.06221


localhost:8502

Select model name

Llama_3_8b

Upload resume PDF:

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

Komal.Patra.pdf 75.3KB

Enter job description:

```
methodologies. (Movement, transformation, and data integrity testing)
Working experience with infrastructure as code, Kubernetes, AWS.
Strong written and oral communication skills
A good attitude and ability to learn quickly on the job
Programming Languages : Python, Spark, SQL
```

Select tone

Professional

Figure 7: Chatbot User Interface

```
{
  "personal_details": {
    "full_name": "Komal  Patra",
    "contact_info": {
      "email": "abc@gmail.com",
      "phone": "+353 00000000",
      "linkedin": "https://www.linkedin.com/in/"
    },
    "professional_summary": "Highly skilled data scientist with
       2.8+ years of experience in data analysis, statistical
       modeling, and data visualization. Proficient in SQL,
       Python, Power BI, and Tableau. Proven expertise in
       leveraging analytical skills to drive business growth and
       improve operational efficiency. Expertise in statistical
       analysis, data visualization, and data storytelling.
       Strong understanding of business objectives and technical
       capabilities."
  },
  "education": [
    {
      "institution": "National College of Ireland",
      "degree": "Master of Science, Data Analytics",
      "field_of_study": "Data Analytics",
      "graduation_date": "09/2023 - 11/2024"
    },
    {
      "institution": "University of Mumbai",
      "degree": "Bachelors of Science in Information Technology",
      "field_of_study": "Information Technology",
      "graduation_date": "08/2016 - 11/2020"
    }
  ],
  "work_experience": [
    {
      "company": "XYZ",
      "title": "Data Scientist Intern",
      "duration": "05/2024 - 08/2024",
      "description": "Designed and deployed innovative machine
         learning solutions using advanced LLM models (T5, BERT,
         RoBERTa) to improve AI-driven customer interactions,
         achieving a 20% increase in chatbot accuracy.",
      "notable_contributions": [
        "Developed and fine-tuned custom LLM models to enhance
           chatbot responses, resulting in a 20% increase in
           accuracy.",
        "Collaborated with cross-functional teams to identify
           business requirements and deliver data-driven
           insights.",
        "Utilized Retrieval-Augmented Generation (RAG)
           techniques to improve contextual understanding and
           response generation of AI systems.",
        "Designed and implemented a data pipeline to integrate
           chatbot interactions with customer feedback, enabling
           data-driven decision making."
      ]
    },
    {
      "company": "Capgemini India Private Limited",
      "title": "Senior Analyst (Digilytics Analyst)",
      "duration": "11/2020 - 08/2023",
      "description": "Analyst for consumer health,
         pharmaceuticals, and crop science websites using SQL,
         Google Analytics, Google, Python, and Power BI.",
      "notable_contributions": [
        "Designed and maintained data architecture for SQL-based
           projects, ensuring seamless data processing and
           accurate reporting across multiple platforms,
           including SSIS and PowerBI.",
        "Managed multiple projects simultaneously with tight
           deadlines and no escalation.",
        "Application Manager for the smart authentication
           project, Successfully migrated all 3 application
           servers into Cloud infrastructure, reducing downtime
           by 30%.",
        "Developed and implemented data quality checks to ensure
           data accuracy and integrity, resulting in a 15%
           reduction in data errors.",
        "Collaborated with cross-functional teams to identify
           business requirements and deliver data-driven
           insights."
      ]
    }
  ],
  "skills": [
    "SQL",
    "Python",
    "Bash scripting",
    "PostgreSQL",
    "MySQL",
    "SQL Server",
    "SSIS",
    "Power BI",
    "Tableau",
    "Excel",
    "Statistical Analysis",
    "Data Visualization",
    "Data Storytelling",
    "Machine Learning",
    "Artificial Intelligence",
    "Retrieval-Augmented Generation (RAG)",
    "T5",
    "BERT",
    "RoBERTa",
    "Advanced LLM models"
  ],
  "certifications": [],
  "publications": [],
  "awards": [],
  "additional_sections": {
    "volunteer_experience": [],
    "languages": [],
    "interests": []
  }
}
```

6 Evaluation

Evaluating the performance of LLM & RAG applications is very challenging as compared to the traditional machine learning approaches. The ground truth for ML is available to evaluate but difficult to exist in LLM application as it consists of various complex tasks. Various techniques are used to evaluate the performance such as Embedding-based metrics, RAGAs metrics, n-gram based metrics and the custom metrics specifically used to evaluate the tailored resume applications. Currently, there is no evaluation technique that is accepted universally due to its complexity often leading to inconsistent evaluation scores. So, several types of techniques are used as per the specific domain tasks by experiment with different hyperparameters, embedding models and data processing strategies. The major difference between using BERT score metrics and RAGAS metrics is that the BERT score calculates the similarity between the reference and generated text using the embedding while in case of RAGAs, the LLM Models are used to evaluate the similarity and the performance of the application.

6.1 Embedding-based Metrics with varying chunk -size hyper-parameter (Experiment 1)

Evaluating the embedding-based metrics using BERTScore reveals that both BM25 and Cross Encoder improve precision, recall, and F1 scores with an increase in chunk size and after using conversational memory refinement which can illustrated from Figure 8. BM25 enhancement boosts the F1 score up to 89% , while the Cross Encoder reaches 87%. BM25 exhibits a minor lead in precision gain after re-ranking; however, the Cross Encoder maintains a balanced performance across metrics. Both approaches demonstrate efficient re-ranking following refinement. As the chunk size varies, the model is improving by identifying the relevant context but with slightly decreased recall. By keeping on refining the model, the slight improvement can be visible across all other metrics in BERTScore which suggest that the model keeps on becoming more accurate by identifying more relevant items without sacrificing recall.

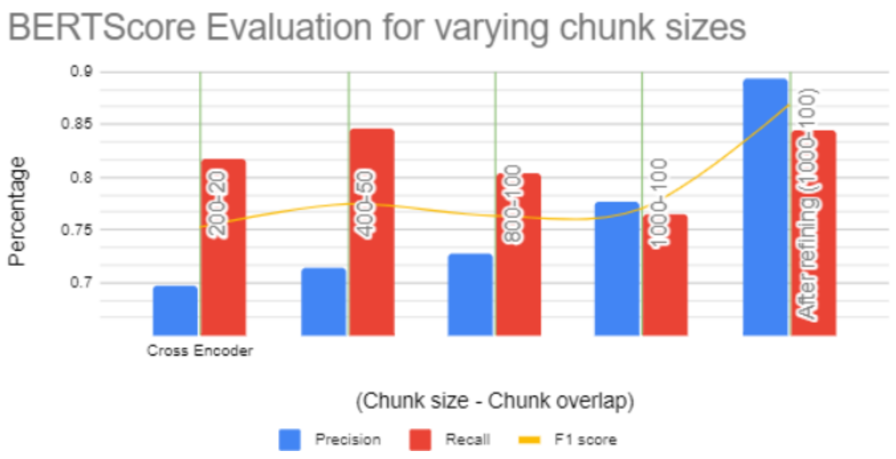
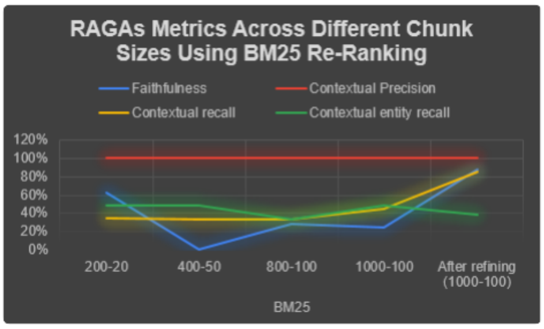


Figure 8: Impact of BERTscore Metrics on chunk sizes for Re-Ranking technique

6.2 Impact of RAGAs metrics variations relative to chunk size for Re-ranking (Experiment 2)

Smaller chunk size will allow the model to focus on a very small specific section to capture the small details which improves the precision but can lead to loss in coherence/Faithfulness, while using bigger chunks helps the model to retrieve and then generate the contextual relevance output. The Chunk size is the essential hyperparameter to study about the RAG system. For BM25, The Faithfulness for the small chunk size starts at 63%, which further varies to 0%. For Cross Encoder, Faithfulness is generally low for all chunk sizes. The BM25 initially shows contextual recall from 25% to 50%, this happend due to variability. As the RAGAs evaluation is been performed using the LLM model itself due to which we cannot rely entirely unless the universally evaluation methods comes into picture.

In contrast, the choice between BM25 and cross encoders depends on the specific metrics and application requirements. According to the evaluation in this study, BM25 proves to be more reliable, particularly in scenarios involving refinement.



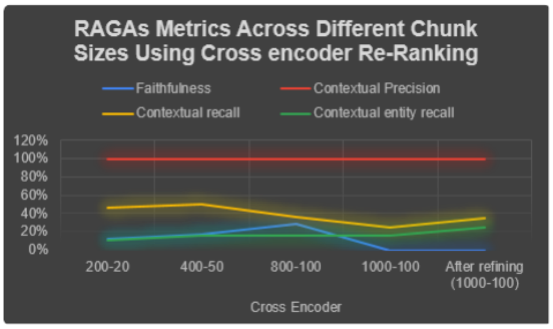


Figure 9: Impact of chunk size on RAGAs Metrics

6.3 Evaluation of Job Alignment and Content Preservation (Experiment 3)

The Job Alignment is a measure that estimates the proximity of the resume and job specifications by assessing both the token and latent spaces similarity. The Content Preservation is meant to evaluate the system's ability to preserve the content of the user's own resume in the generated on so as to identify and solve any arising problems. This metric provides an overview of how well the created resumes preserves the essence of the original resume specified by the user.

The overlap coefficient for both the custom metrics calculates the overlapping keywords between the two sets specifically focusing on smaller sets. The Jaccard similarity calculates the similarity between the intersection of two different sets where 0 is representing no similarity while 1 represents the complete similarity. The BM25 shows the improvement in all the evaluating similarity metrics for both Job alignment and content preservation metrics after refinement, specifically overlap coefficient and the cosine similarity. But the cross encoder outperforms compared to BM25 by achieving high scores in Job alignment metrics as our main focus is to tailor the resume based on job specifications. However, the content preservation metrics is bit similar with some metrics decreasing after refinement.

In Table 2 and Table 3, the evaluation score for the Job alignment and content preservation is provided.

Table 2: Comparison of BM25 Re-ranking techniques Across Different Stages and Metrics

Metric	**Job Alignment**	**Initial Tailoring of Resume**	**Refined Tailored Resume**
Job Alignment	Overlap Coefficient	0.169	0.187
	Jaccard Similarity	0.066	0.070
	Cosine Similarity	0.151	0.539
Content Preservation	Overlap Coefficient	0.750	0.969
	Jaccard Similarity	0.520	0.726
	Cosine Similarity	0.690	0.890

Table 3: Comparison of Cross Encoder Re-ranking technique Across Different Stages and Metrics

Metric	**Job Alignment**	**Initial Tailoring of Resume**	**Refined Tailored Resume**
Job Alignment	Overlap Coefficient	0.183	0.224
	Jaccard Similarity	0.070	0.0837
	Cosine Similarity	0.523	0.524
Content Preservation	Overlap Coefficient	0.900	0.909
	Jaccard Similarity	0.683	0.632
	Cosine Similarity	0.872	0.864

6.4 Discussion

In this comparative study, the RAG was implemented with the LLM model to experiment with relevancy and accuracy along with the custom evaluation metrics. During the implementation, various type of Re-ranking metrics such as keyword search (BM25) and vector search is experimented with to analyse and improve the performance of an application.Thus, the Problem emerges when choosing the best optimal hyperparameters to enhance the generated resume as they are completely dependent on them. In this research, the max token limit was the problem encountered because the model was provided with extensive larger inputs, consisting of the job description, the candidate's resume and the top 10 relevant documents retrieved from the vector database during the process. Utilising the model directly without the assistance of ChatGroq required a lot of computational power, particularly a powerful GPU infrastructure, which was essential to effectively process the data. While using ChatGroq, several models were restricted to the limit due to which it was not feasible to use those models. The hallucination score was almost around 0.1890 which refers the vary low hallucinations, it means that the model consider to be relevant and consisting of ground truth. So the variants of the LLAMA model can directly impact the performance. The cosine similarity for the content preservation when evaluated for the custom metrics was found to be 72% while using RAG it was 89% which decreases the hallucinations. Therefore the RAG with LLM outperform compared to rely only on LLM.

7 Conclusion and Future Work

To achieve the objectives of the project, the research focused on developing tailored resumes based on job description to solve the challenges faced by job seekers. The dataset contains resume from diverse domains to enhance the probability of retrieving the relevant and similar document with high ranking score. The resume was parsed into JSON format by extracting the information as per the sections by using the LLM models. The ChatGroq LLAMA-3-70B model performed well for information retrieval with less inference time. The RAG was integrated with LLM were the BM25 (keyword-search) re-ranking technique worked well to provide a better-generated response. The chatbot has been developed using Streamlit along with implementing the conversational memory for better user interface experience. Further, the tailored resume was evaluated using several type of performance metrics such as embedding-based, LLM-based and custom metrics such as Job Alignment and Content preservation. However, the development of a reliable technique to evaluate the performance metrics for LLM and RAG is still under research to provide the accurate and consistent result.

In future, the gaps can be addressed by focusing on creating the more precise and standarised custom metric only designed specifically to evaluate the performance aligning specifically for this task. Additionally, Future efforts could explore implementing the knowledge graph or RAFT (combination of RAG and Fine-tuning) to further enhance the performance and accuracy of these models.

References

Balaji, J., Sigdel, M., Hoang, P., Liu, M. and Korayem, M. (2019). Airesume: Automated generation of resume work history., *KaRS@ CIKM*, pp. 19–25.

Bruera, A., Alda, F. and Di Cerbo, F. (2022). Generating realistic synthetic curricula vitae for machine learning applications under differential privacy, *in* I. Siegert, M. Rigault and V. Arranz (eds), *Proceedings of the Workshop on Ethical and Legal Issues in Human Language Technologies and Multilingual De-Identification of Sensitive Data In Language Resources within the 13th Language Resources and Evaluation Conference*, European Language Resources Association, Marseille, France, pp. 53–63.
URL: *https://aclanthology.org/2022.legal-1.11*

Foulds, P. F., James, R. and Pan, S. (2024). Ragged edges: The double-edged sword of retrieval-augmented chatbots, *arXiv preprint arXiv:2403.01193* .

Hanna, M. and Bojar, O. (2021). A fine-grained analysis of bertscore, *Proceedings of the Sixth Conference on Machine Translation*, pp. 507–517.

Kale, S. S. and Andreopoulos, W. B. (2023). Job tailored resume content generation, *2023 IEEE Ninth International Conference on Big Data Computing Service and Applications (BigDataService)*, IEEE, pp. 40–47.

Lewis, P. S. H., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S. and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks, *CoRR* **abs/2005.11401**.
URL: *https://arxiv.org/abs/2005.11401*

Li, H., Su, Y., Cai, D., Wang, Y. and Liu, L. (2022). A survey on retrieval-augmented text generation.
URL: *https://arxiv.org/abs/2202.01110*

Sunico, R. J., Pachchigar, S., Kumar, V., Shah, I., Wang, J. and Song, I. (2023). Resume building application based on llm (large language model), *2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pp. 486–492.

Yu, H., Gan, A., Zhang, K., Tong, S., Liu, Q. and Liu, Z. (2024). Evaluation of retrieval-augmented generation: A survey, *arXiv preprint arXiv:2405.07437* .

Zhao, P., Zhang, H., Yu, Q., Wang, Z., Geng, Y., Fu, F., Yang, L., Zhang, W., Jiang, J. and Cui, B. (2024). Retrieval-augmented generation for ai-generated content: A survey.
URL: *https://arxiv.org/abs/2402.19473*

Zinjad, S. B., Bhattacharjee, A., Bhilegaonkar, A. and Liu, H. (2024). Resumeflow: An llm-facilitated pipeline for personalized resume generation and refinement, *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2781–2785.