



Tunisian Republic
Ministry of Higher Education and Scientific Research
Carthage University - Engineering School of Statistics and Information Analysis

Submitted to

Engineering School of Statistics and Information Analysis

In partial fulfillment of the requirements for the degree of

National Engineering Diploma in Statistics and Information Analysis

by

Saif Mechi

Elevating Resume Screening with Large Language Models

Defended on 08/10/2024 in front of the committee composed of:

Ms Sellami HAJER	President
Ms Hamdeni TASNIME	Reviewer
M. Bel Mufti GHAZI	Supervisor
M. Afli JAWHAR	Mentor
M. Benhaj. M MUSTAPHA	Co-Mentor

A Graduation Project made at



Dedication

*This work is dedicated to my loving family,
whose constant support and encouragement have been my guiding light throughout this journey.
To my parents, for their endless love and belief in my potential, and to my siblings, for their
companionship and understanding.*

*A special dedication goes to my uncle,
whose wisdom and guidance have inspired me to pursue my dreams with determination and
integrity. His words of encouragement and the values he instilled in me have been instrumental in
shaping the person I am today.*

*To my dear friends,
thank you for your unwavering support, camaraderie, and for always being there when I needed you
most. Your friendship has made this journey more fulfilling and enjoyable.*

Thank you all for being my pillars of strength.

Thanks

I could never have realized this project without the precious help and support of many people ... First, I would like to thank my academic supervisor, M. Bel Mufti GHAZI for his guidance, encouragement, and insightful feedback throughout this journey. His expertise and support have been instrumental in shaping this work.. I would like to express my gratitude to M. Afli JAWHAR and M. Benhaj. M MUSTAPHA ,for inspiring me to pursue an amazing project within Instadeep . Their advice and support have been vital to the success of this project, and I greatly appreciate their willingness to share their knowledge and experience. I also thank him for his welcome... I would also like to say to jury president Ms Sellami HAJER how honored I am. I am very grateful to Ms Hamdeni TASNIME for making herself available, I am especially grateful to her for the interest she has shown in this project by committing to be a reporter.

My gratitude goes to those who have provided emotional support for this work: my family and friends.

Abstract

*This report presents a research project conducted within the Instadeep as part of the requirements for the National Engineering Diploma at Engineering School of Statistics and Information Analysis . The study aims to enhance **resume screening systems**, focusing on improving the shortlisting process for candidates based on job postings by leveraging the capabilities of **large language models** (LLMs). The research began with a Semantic Search approach using dense vector embeddings. However, this initial method did not fully meet the primary objectives of the task, necessitating further iteration. Through this iterative process, we eventually developed a custom Retrieval-Augmented Generation (RAG) system that incorporates LLMs for both filtering and ranking tasks. This final approach significantly outperformed earlier methods, delivering superior results in key metrics such as Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG). The proposed solution has proven effective across multiple domains and is efficient in terms of cost, speed, and resource utilization.*

Keywords: *Resume Screening, Human Resources Technology, Large Language Models, Retrieval-Augmented Generation, Semantic Search, Transformer, Openai*

Contents

Contents	ii
List of Figures	iii
List of Tables	iv
List of Algorithms	v
1 Generality	2
1.1 Host Company	2
1.2 Problem	3
1.3 Proposed Solution	4
2 Large Language Models	6
2.1 Introduction	6
2.2 Evolution of Language Models	6
2.3 Overview of Large Language Models	7
2.4 Transformer Architecture	8
2.4.1 Encoder-Decoder	9
2.4.2 Attention Mechanism in Transformer	9
2.5 Pretraining of Large Language Models	10
2.6 Large Language Model Variants	11
2.6.1 Embedding Models	11
2.6.2 Generative Models	11
2.7 Evaluation of Large Language Models	12
2.8 Leveraging LLMs in Practice	12
2.9 Retrieval-Augmented Generation (RAG)	14
2.10 LLM Alignment	15
3 Literature Review	17
3.1 Introduction	17
3.2 Manual Resume Screening	17
3.2.1 Structure and Content of Resumes and Job Descriptions	17
3.2.2 Resume Screening Techniques for HR	19
3.2.3 Challenges and Limitations of Manual Screening	20
3.3 Related Work on the Person-Job Fit Problem	21
3.3.1 The Evolution and Definition of Applicant Tracking Systems	21
3.3.2 Approaches to the Person-Job Fit Problem	21

3.3.3	State of the Art Overview	22
3.3.4	Limitations of State-of-the-Art Approaches	24
3.4	Conclusion	24
4	Data Collection and Preparation Pipeline	25
4.1	Introduction	25
4.2	Data Collection and Preparation Pipeline	25
4.3	Data Collection	26
4.3.1	Resume Datasets	26
4.3.2	Metadata Extraction	27
4.3.3	Visualization and Exploratory Analysis	28
4.4	Creation of the Ground Truth Dataset	31
4.5	Evaluation Protocol	33
4.5.1	Stage 1: Evaluating Retrieval Performance	33
4.5.2	Stage 2: Evaluating Ranking Quality	33
4.6	Conclusion	35
5	Experiments	36
5.1	Introduction	36
5.2	Working Environment	36
5.2.1	Software Environment	36
5.2.2	Hardware Environment	37
5.3	Working Methodology	37
5.4	Semantic Search Approach	37
5.4.1	System Setup	38
5.4.2	Similarity Calculation	39
5.4.3	Initial Results	39
5.4.4	Issues Detected	39
5.4.5	Improvements	40
5.4.6	Conclusion	41
5.5	Hybrid Search Approach	41
5.5.1	Hybrid Search vs Semantic Search	43
5.5.2	Robustness Testing with Increased Dataset Sizes	44
5.6	Sequential Search Approach	45
5.6.1	Final Results of the Sequential Search	46
5.7	Cross Encoder Reranking Approach	47
5.8	LLM-Based Reranking	49
5.9	Conclusion	50
	Conclusion	51
	Bibliography	52

List of Figures

- 1.1 InstaDeep Offices Location 3
- 2.1 The Transformer Architecture 8
- 2.2 Components of the Retrieval-Augmented Generation (RAG) system. 14
- 3.1 Overview of the Structure of Resumes and Job Descriptions. 19
- 4.1 Overview of the Data Collection and Preparation Pipeline 26
- 4.2 Confusion Matrix for the labeling of AI jobs. 28
- 4.3 Distribution of job postings across various categories. 29
- 4.4 Distribution of resumes across different categories. 30
- 4.5 Distribution of job postings by experience level 30
- 4.6 Distribution of resumes by experience level. 30
- 5.1 Workflow in the Semantic Search System 38
- 5.2 Massive Text Embedding Leaderboard. 39
- 5.3 Workflow for Hybrid Search Approach 42
- 5.4 Workflow of the Sequential Search Approach 45
- 5.5 Workflow in the Semantic Search System 46
- 5.6 Workflow for Cross Encoder Ranking Approach 47
- 5.7 Workflow for LLM-Based Reranking Approach 49

List of Tables

- 3.1 Comparison of Key Approaches in Person-Job Fit Research 23
- 5.1 Evaluation metrics for Semantic Search 39
- 5.2 Recall Values of Semantic Search 40
- 5.3 Comparison of Recall Scores for Different Encoders at Various Recall Levels. 40
- 5.4 Recall scores for different query approaches. 41
- 5.5 Hybrid Search Results 43
- 5.6 Comparison: Hybrid Search and Semantic Search 43
- 5.7 Performance of Hybrid Search with Augmented Sample Sizes. 44
- 5.8 Recall Performance at Different Sample Sizes 45
- 5.9 Performance of Sequential Search vs. Random Baseline 47
- 5.10 Performance Metrics for Various Approaches. 48
- 5.11 Comparision Between all Approaches. 49

List of Algorithms

1 [BM25 Algorithm](#) 42

General Introduction

Effective recruitment of talented and suitable candidates is a cornerstone of any organization's success and competitiveness. The ability to attract and select individuals who possess the right skills and qualifications directly impacts the organization's performance, innovation, and overall growth. Given the increasing competition in the job market, organizations must adopt rigorous and efficient methods to identify and hire the best candidates [9]. One way to accomplish this is by refining the processes used to screen and evaluate potential hires.

However, in today's fast-paced job market, the challenge of accurately matching candidates with job descriptions is akin to finding a needle in a haystack. Traditional manual resume screening, although widely practiced by HR professionals, is both time-consuming and labor-intensive. This often leads to delays and inefficiencies in the hiring process. According to recent statistics, global job openings reached an estimated 192 million in 2023 [9], with an average of 118 applicants per advertised position [4]. This overwhelming volume underscores the difficulty for hiring teams to identify the most suitable candidates and for applicants to distinguish themselves from the crowd.

Here comes the need for a modern solution that can streamline and enhance the recruitment process. The rise of large language models (LLMs) has introduced transformative advancements in natural language processing. These models, with their remarkable capacity to understand and generate human language, have significantly enhanced our ability to capture and interpret textual nuances. This technology presents an opportunity to improve traditional resume screening methods, making them more efficient and accurate.

In this context, our research focuses on the potential of LLMs to revolutionize resume retrieval systems. By leveraging the advanced capabilities of LLMs in text embedding and generation, we propose a solution designed to better align and rank resumes with job descriptions, thereby optimizing the recruitment process. This integration of LLMs is expected to address the limitations of traditional methods and offer a more effective approach to candidate selection.

The structure of this report is carefully crafted to provide a comprehensive overview of the research undertaken. It begins with an examination of the general context and significance of the problem. Following this, a detailed discussion on the rise and capabilities of LLMs is presented. The report then includes a dedicated chapter on data collection and preparation, outlining how data was sourced, cleaned, and processed to support the research. Afterward, the proposed solution, which utilizes LLMs for an enhanced candidate shortlisting system, is introduced. Subsequently, the methods employed and the results obtained are discussed. Finally, the report concludes with an analysis of the effectiveness of the proposed approach and its broader implications.

Chapter 1

Generality

1.1 Host Company

This project was conducted at InstaDeep, a leading AI firm founded in 2014 by Karim Beguir and Zohra Slim. Over the years, InstaDeep has grown into an established entity within the AI industry, boasting a workforce of over 400 employees. The company’s headquarters is based in London, with additional offices located in Paris, Tunis, Lagos, Cape Town, and Dubai.

InstaDeep specializes in delivering AI-powered decision-making systems for enterprises, effectively bridging the gap between cutting-edge machine intelligence research and practical business applications. Their expertise spans various domains, including machine learning, deep learning, and particularly reinforcement learning—an area where systems learn and improve through experience. This technological prowess enables InstaDeep to provide its clients with a significant competitive edge in today’s AI-driven landscape.

The firm’s capabilities are not confined to a single industry but extend across multiple sectors, including logistics, mobility, energy, and electronic design. By harnessing the power of GPU-accelerated computing, deep learning, and reinforcement learning, InstaDeep has developed advanced AI systems capable of addressing some of the most complex challenges these industries face.

In 2023, InstaDeep’s influence and global footprint expanded further when it became part of the BioNTech Group, an organization that shares InstaDeep’s innovative culture and vision for leveraging AI for societal good. This strategic alignment underscores InstaDeep’s commitment to democratizing AI, fostering an inclusive future, and nurturing talent to contribute meaningfully to the global AI discourse.



Figure 1.1: InstaDeep Offices Location

1.2 Problem

The primary objective of this project is to develop an advanced recruitment system that effectively mimics the decision-making abilities of human HR professionals, particularly in the initial stages of resume screening and candidate evaluation. The core challenge lies in harnessing the power of Large Language Models (LLMs) to accurately and efficiently rank candidates to a given job and across diverse domains. This task requires a deep understanding of the nuances in job requirements and candidate qualifications, ensuring the system can make informed and context-aware decisions to shortlist the most suitable candidates.

One of the significant challenges encountered in this project is adapting LLMs to the specific task of resume screening. While LLMs excel at natural language processing, they are not inherently designed for recruitment tasks. Therefore, the project must address the challenge of adapting these models to interpret and evaluate resumes in a manner that aligns with HR professional's expectations. This involves developing a system that can assess qualifications, experience, and skills with a level of accuracy that matches human judgment while also ensuring that the model remains adaptable to the dynamic nature of the job market.

Another critical aspect of the problem is addressing the diverse staffing needs of enterprises, which often require personnel for a wide range of roles and positions spanning different domains. In such environments, a recruitment system must be capable of effectively managing the hiring process across multiple industries and job types. The system should be adaptable enough to understand the specific requirements of various roles while maintaining high accuracy in candidate matching.

Additionally, the system must be efficient not only in terms of accuracy but also in terms of cost and time. The recruitment process is often time-sensitive, with organizations seeking to fill positions as quickly as possible to avoid disruptions in operations. Therefore, the proposed solution must be designed to streamline the recruitment process, significantly reducing the time required to screen

resumes and identify the best-fit candidates. At the same time, the system should be cost-effective, offering a viable alternative to traditional, labor-intensive recruitment methods.

In summary, the problem we aim to address involves the development of an effective, non-domain-specific recruitment system that leverages LLMs to mimic human HR screening abilities. This system must be capable of adapting to the dynamic nature of the job market, ensuring efficiency in terms of both time and cost, while maintaining a high level of accuracy and fairness in candidate selection.

1.3 Proposed Solution

To overcome the challenges identified in the recruitment process, this project leverages the capabilities of Large Language Models (LLMs) to build an advanced system for resume screening and candidate matching. Unlike traditional AI-based methods that rely heavily on static rule-based algorithms or keyword matching, our solution is driven by the deep understanding of LLMs, enabling the system to accurately process and understand the complexities of the task.

The power of LLMs lies in their ability to comprehend both explicit and nuanced information embedded in human language. These models can extract not only specific skills, qualifications, and experiences but also capture context, relevance, and implied competencies—critical factors in evaluating resumes. By using multiple LLM components throughout the pipeline, our system ensures robust candidate evaluations across different stages of the screening process.

One of the key features of the proposed solution is its domain-agnostic design, it works across different industries and job roles without needing special adjustments. Unlike traditional AI systems that often require extensive customization for each new domain, our system uses LLMs that can handle various job types—whether they’re technical, managerial, or creative—without extra training. This means it’s highly adaptable and can be used effectively in many different sectors and job markets.

Additionally, the efficiency of this solution is crucial. The system is designed to process resumes quickly and effectively while using minimal GPU resources. It’s built to streamline the resume screening process, reducing the time and computational power required. By leveraging LLMs, the system can swiftly evaluate resumes and match them to job descriptions, mimicking the depth and understanding of human recruiters. This approach ensures that the most relevant candidates are identified efficiently, with less reliance on extensive computational resources compared to other related works.

Another significant aspect of our proposed solution is that it is built to handle the dynamic nature of the job market. The skills and qualifications required for jobs are constantly evolving, and the LLM-based system is capable of understanding and integrating new trends without requiring manual updates or static rule changes.

In conclusion, by integrating LLMs into various stages of the recruitment process, our system offers a highly adaptable, efficient, and scalable solution to the challenges of modern recruitment. It improves the accuracy and speed of resume screening, while its domain-agnostic design and adaptability ensure it remains relevant across industries and as the job market evolves. This innovative use of LLM technology positions the system as a valuable tool for organizations looking to enhance their recruitment processes and outcomes.

Conclusion

This chapter has provided an overview of the host company, InstaDeep, and outlined the context and challenges associated with our recruitment system project. We have discussed the need for a versatile, domain-agnostic solution to effectively handle diverse job roles and industries while optimizing resource use. The general approach to our solution involves leveraging the capabilities of Large Language Models (LLMs) to enhance the efficiency and accuracy of resume screening. In the following chapter, we will delve into the field of LLM, exploring its applications and historical developments. This exploration will set the stage for a deeper understanding of how LLM technologies contribute to our proposed solution and the broader landscape of recruitment systems.

Chapter 2

Large Language Models

2.1 Introduction

In this chapter, we will begin by providing an overview of Large Language Models (LLMs), highlighting their significance and key applications in natural language processing. We will then move on to the core architecture of LLMs, exploring the structural foundations that enable their advanced capabilities. Following this, we will examine different variants, and how they serve specific tasks. Finally, we will discuss practical approaches for leveraging LLMs, including Retrieval-Augmented Generation (RAG), and methods to align LLMs with specific tasks or domains for optimal performance.

2.2 Evolution of Language Models

Language models (LMs) are a foundational component of natural language processing (NLP), designed to predict the likelihood of a sequence of words based on preceding context. These models are integral to a wide range of NLP tasks, including text generation, machine translation, and information retrieval. The evolution of language models can be categorized into four major stages[29], each representing significant advancements in their ability to understand and process language.

- **Statistical Language Models (SLMs):** Emerging in the 1990s, SLMs relied on the Markov assumption to predict the next word in a sequence based on a fixed number of preceding words. Known as n-gram models, they focused on short-term dependencies, such as bigrams (two-word sequences like "I am") and trigrams (three-word sequences like "All is well"). While SLMs were widely used for tasks such as information retrieval and basic language understanding, they struggled with modeling long-term dependencies due to data sparsity and the exponential increase in transition probabilities as more words were considered. Techniques like backoff and Good-Turing estimation were developed to address these challenges, but the models remained limited in their ability to capture broader language patterns.
- **Neural Language Models (NLMs):** To overcome the limitations of statistical approaches, neural networks were introduced, leading to the development of NLMs. These models, including architectures like multilayer perceptrons (MLPs) and recurrent neural networks (RNNs), marked a significant shift by focusing not just on word order, but on the deeper relationships between words. The introduction of word embeddings, such as word2vec, allowed these models to represent words as dense vectors in a high-dimensional space, capturing semantic meaning

beyond simple word frequency. This transition enabled NLMs to handle more complex tasks, such as generating coherent text and performing machine translation, with greater success than their statistical predecessors.

- **Pre-trained Language Models (PLMs):** Pre-trained language models represent a major advancement in NLP. These models are trained on vast amounts of text data through unsupervised learning, allowing them to develop a deep understanding of language before being fine-tuned for specific tasks. Early PLMs like ELMo (Embeddings from Language Models) and BERT (Bidirectional Encoder Representations from Transformers) introduced the concept of context-aware word representations, where the meaning of a word is informed by its surrounding context. This breakthrough enabled significant improvements in NLP tasks such as text classification, sentiment analysis, and question answering. The pre-training and fine-tuning paradigm established by these models laid the groundwork for more sophisticated models like GPT-2 and BART.
- **Large Language Models (LLMs):** The advent of LLMs marks the latest stage in the evolution of language models, characterized by a substantial increase in model size and the amount of training data. LLMs, such as GPT-3 and PaLM, are capable of performing a wide range of complex tasks, often demonstrating emergent abilities that were previously thought to be beyond the reach of language models. These models leverage the power of transformers and extensive training corpora to achieve remarkable accuracy in tasks like text generation, conversational AI, and summarization. LLMs, exemplified by systems like ChatGPT, have become central to modern NLP, setting new benchmarks for language understanding and generation.

In the following sections, we will explore Large Language Models (LLMs) in greater depth, focusing on their architecture, capabilities, and the revolutionary impact they have had on the field of natural language processing.

2.3 Overview of Large Language Models

Large Language Models (LLMs) represent a new and advanced type of language model built using state-of-the-art deep learning architectures. These models are trained on massive corpora of text data, enabling them to understand and generate human language with high precision. By leveraging sophisticated techniques in deep learning, LLMs are capable of handling a wide range of language-related tasks, from comprehending complex language structures to predicting text and generating responses that closely mimic human communication. Their extensive training allows them to achieve remarkable accuracy and fluency in various applications.

What sets LLMs apart is their ability to excel at tasks that were once incredibly challenging for machines. Whether it's translating languages, summarizing long documents, answering detailed questions, or even engaging in meaningful conversations. Their capabilities have made a significant impact across industries, powering everything from virtual assistants and customer service bots to content generation, legal analysis, and healthcare insights.

LLMs are more than just tools for understanding text. They represent a leap forward in AI's ability to interact with the world through language, automating tasks that once required human expertise.

Their versatility and efficiency have opened up new possibilities, making them an essential part of the future of AI, with applications that extend far beyond traditional language tasks.

2.4 Transformer Architecture

The Transformer model, introduced by Vaswani et al. [27], is a key architectural innovation in the development of Large Language Models (LLMs). Unlike previous models relying on recurrent or convolutional neural networks. As shown in Figure 2.1, the Transformer leverages an encoder-decoder architecture fully driven by self-attention mechanisms. This architecture allows for efficient parallelization and long-range dependency modeling, crucial for the effective training and performance of LLMs.

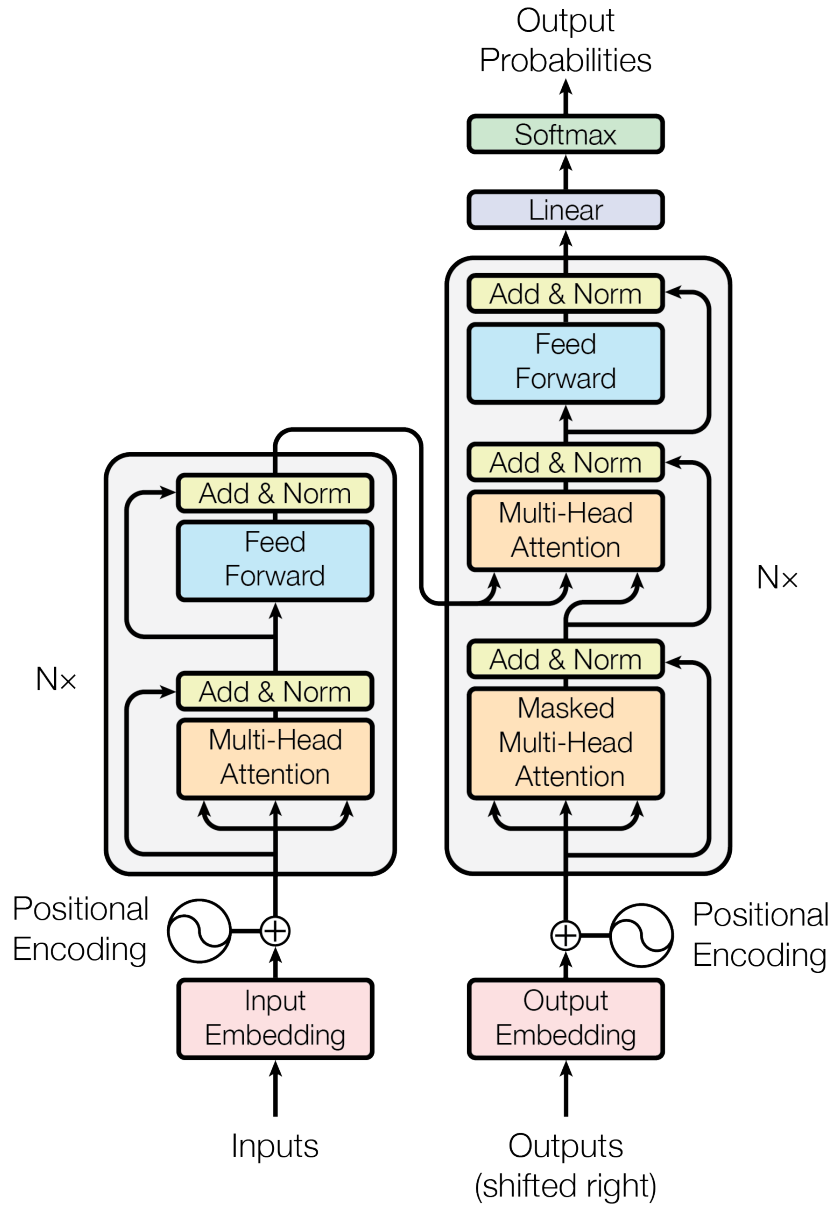


Figure 2.1: The Transformer Architecture .

2.4.1 Encoder-Decoder

The **encoder** represented by the left block in figure 2.1 is responsible for processing an input sequence, such as a sentence or paragraph, by converting it into a series of hidden representations. These representations encapsulate both the semantic and positional relationships between tokens (words or subwords) in the input sequence. The encoder consists of multiple layers, each comprising two main components: a multi-head self-attention mechanism and a position-wise feed-forward network. The self-attention mechanism allows each token to attend to all other tokens in the sequence, enabling the model to capture both local and global dependencies simultaneously, while the feed-forward network adds additional complexity, helping the model learn more intricate patterns.

The **decoder**, on the other hand, represented by the right block in figure 2.1 generates the output sequence, conditioned on both the encoder's output and the previously generated tokens in the target sequence. Unlike the encoder, the decoder uses masked self-attention mechanisms to attend to the target sequence, preventing tokens from attending to future positions, along with an additional encoder-decoder attention mechanism. This attention mechanism allows the decoder to focus on relevant parts of the input sequence during the generation process. The decoder operates autoregressively, meaning that it generates each token step-by-step, attending to previously generated tokens and the entire input sequence.

2.4.2 Attention Mechanism in Transformer

The core component of the Transformer architecture is its attention mechanism, which replaces the recurrence seen in earlier models like RNNs[21] and LSTMs[8]. Unlike these earlier models, the attention mechanism allows the Transformer to process the entire input sequence simultaneously, rather than sequentially. This is done by computing three vectors for each token in the sequence:

- **Query (Q) vector:** This vector represents the token that is seeking information. It essentially asks the question: "Which other tokens in the sequence should I pay attention to?"
- **Key (K) vector:** This vector corresponds to the token that provides information. It acts as the answer to the query, helping identify which tokens are relevant in response to the query.
- **Value (V) vector:** This vector holds the actual information that will be passed along if the query and key vectors deem it important. It contains the data that the model ultimately uses for further computations.

The attention mechanism computes how much attention each token should give to others by calculating the dot product between the query vector of one token and the key vector of another. This produces a score, which is then used to weigh the value vectors, determining how much influence each token has on the others within the sequence.

The self-attention mechanism is responsible for computing the relevance of each token to every other token, capturing contextual information from across the entire sequence. The attention score is calculated as the dot product of the query and key vectors, followed by a scaling operation. The softmax function is then applied to obtain a normalized attention weight for each token, which is subsequently used to compute a weighted sum of the value vectors.

The scaled dot-product self-attention is formally defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.1)$$

where Q represents the query matrix, K is the key matrix, V is the value matrix, and d_k is the dimensionality of the keys. The scaling factor $\frac{1}{\sqrt{d_k}}$ is applied to prevent the dot products from becoming too large, which can negatively affect the gradients during training.

This mechanism is applied in both the encoder and decoder of the Transformer architecture, enabling the model to efficiently capture dependencies between tokens without the sequential constraints imposed by recurrent networks. By leveraging attention, the Transformer is able to model long-range dependencies and handle large input sequences in parallel, making it highly scalable for training on vast datasets.

In summary, the combination of the encoder-decoder framework with the self-attention mechanism forms the backbone of the Transformer model. This architecture has proven essential in the training and scalability of large language models (LLMs), allowing them to capture intricate language patterns and dependencies over extended contexts.

2.5 Pretraining of Large Language Models

The pretraining of large language models (LLMs) builds on the Transformer architecture, which is fundamental to their ability to process and understand complex language patterns. This phase involves training the model using extensive text corpora, enabling it to acquire a broad and nuanced understanding of language.

Training LLMs requires exposing the model to diverse and vast datasets. Major sources include the Common Crawl dataset¹, which consists of text from a wide array of web pages, Wikipedia for structured and factual information, and the Books Corpus, which offers a variety of literary texts. These datasets ensure that the model learns from a wide range of contexts and language styles, contributing to its robustness.

Self-supervised learning objectives are central to the pretraining process. Two key objectives are commonly used: Masked Language Modeling (MLM) and Causal Language Modeling (CLM). In MLM, random tokens in a text are masked, and the model learns to predict these masked tokens based on their surrounding context. This method helps models like BERT[5] understand context and relationships within text. On the other hand, CLM, used by models like GPT[19], focuses on predicting the next token in a sequence, facilitating coherent text generation by learning from preceding tokens.

The training process is computationally intensive, often requiring distributed computing resources such as multiple GPUs or TPUs. Efficient training involves careful selection of hyperparameters and optimization techniques to ensure that the model learns effectively from the data.

Once pretraining is complete, the LLM is ready for inference, meaning it can perform various language-related tasks such as text generation, translation, and question-answering. The pretrain-

¹The Common Crawl corpus contains raw web page data, metadata extracts, and text extracts. It has been growing steadily at 200-300 TB per month for the last few years.

ing phase equips the model with the necessary capabilities to handle these tasks by leveraging the extensive knowledge it has gained.

In summary, the pretraining of LLMs involves utilizing the Transformer architecture, massive datasets, and unsupervised learning objectives to develop models that are adept at understanding and generating human language.

2.6 Large Language Model Variants

Large Language Models (LLMs) can be broadly categorized based on their underlying architecture into generative and embedding models. Each type serves distinct purposes in natural language processing (NLP) tasks.

2.6.1 Embedding Models

Embedding models are the models that are trained only with the encoder part of the transformer and they focus exclusively on understanding and representing input sequences. These models process text input through multiple layers of self-attention mechanisms, generating a sequence of hidden states that encapsulate the meaning and context of the text. The self-attention mechanism allows the model to weigh the relevance of different parts of the input sequence when producing the output representation.

A prominent example of an encoder-only model is BERT (Bidirectional Encoder Representations from Transformers) [5]. BERT is particularly effective for tasks requiring a deep understanding of the input text. Key applications include:

- **Text Classification:** Assigning categories to text, such as sentiment analysis or topic labeling.
- **Named Entity Recognition (NER):** Identifying entities such as names, dates, and locations within the text.
- **Question Answering:** Extracting answers from a passage of text based on a given query.

BERT is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right contexts in all layers.

2.6.2 Generative Models

Generative models are the models trained with the decoder part of the transformer, in contrast, are primarily focused on generating text. These models predict the next token in a sequence based on the context provided by previous tokens, with the objective of minimizing the negative log likelihood of the predicted token. They are well-suited for tasks where generating coherent and contextually relevant text is critical.

A notable example of a decoder-only model is the GPT (Generative Pre-trained Transformer) series [19]. GPT models excel in:

- **Text Generation:** Creating coherent and contextually appropriate text, such as for chatbots or content creation.
- **Text Completion:** Completing sentences or paragraphs based on a given prompt.

- **Conversational AI:** Engaging in dialogue with users in a natural and human-like manner.

GPT models are autoregressive, meaning they predict each word in a sequence one at a time, conditioning on the previously generated tokens.

Both encoder-only and decoder-only models have unique strengths, making them suited for different types of NLP tasks. Understanding these distinctions is crucial for selecting the appropriate model for specific applications.

2.7 Evaluation of Large Language Models

Evaluating large language models (LLMs) involves both quantitative and qualitative methods, each contributing valuable insights into the model's performance and practical utility.

Quantitative evaluation focuses on numerical metrics that provide objective insights into the model's performance. Key metrics include accuracy, which measures the proportion of correct predictions; precision and recall, which together assess the model's ability to retrieve relevant information and identify all relevant instances; and the F1 score, which balances precision and recall into a single metric. Additionally, benchmark metrics such as the General Language Understanding Evaluation (GLUE) aggregate scores across various tasks to offer a broad measure of the model's language understanding capabilities [28].

Qualitative evaluation explores aspects of model performance that are not captured by numerical metrics alone. This assessment includes examining the model's robustness to diverse inputs, fairness in producing equitable outcomes, interpretability of its decision-making process, and contextual understanding of complex scenarios. Benchmarks such as the Holistic Evaluation of Language Models (HELM) and methods like AlpacaFarm address these aspects by evaluating models across a range of scenarios and metrics, including accuracy, calibration, robustness, fairness, and bias [14, 6]. These evaluations are critical for understanding the practical utility and societal impact of LLMs beyond what is revealed through quantitative measures alone.

2.8 Leveraging LLMs in Practice

As discussed in Section 2.6, LLMs can be broadly categorized into two primary architectures: encoder-only and decoder-only, each designed to address different types of tasks. These models play pivotal roles in applications such as embedding, transfer learning, and prompting, offering flexible and powerful solutions to complex language problems.

This section explores how these models are utilized in tasks such as embedding, transfer learning, and prompting.

Encoder-Only Models: Embedding and Transfer Learning

Encoder-only models, such as BERT, excel in generating high-quality embeddings that capture the semantic meaning of text. These embeddings are dense vector representations that allow for efficient comparison and categorization of text, making them invaluable in tasks such as text classification, clustering, and information retrieval. The encoder's ability to create context-aware embeddings facilitates accurate and nuanced understanding of text inputs.

In addition to embedding, encoder-only models are also heavily utilized in transfer learning. Transfer learning involves adapting a pre-trained model to a smaller, domain-specific dataset, often by freezing all of the pre-trained weights to retain the broad knowledge acquired during initial training, while adding task-specific layers to tailor the model to the current task. This approach enables efficient use of computational resources and data, particularly in domains where labeled data may be scarce.

Decoder-Only Models: Prompting for Text Generation

Decoder-only models, such as GPT, are particularly well-suited for text generation tasks, which are often driven by prompt engineering. Prompting involves carefully crafting input prompts to guide the model's output towards generating desired text. This technique leverages the model's ability to produce coherent and contextually relevant language based on the given prompt, making it effective for a wide range of generative tasks, including content creation, dialogue systems, and automated reporting.

The power of decoder-only models lies in their autoregressive nature, which enables them to predict the next word in a sequence based on the preceding context. This allows for the generation of fluent and human-like text, tailored to specific requirements as dictated by the prompt. Through prompting, users can harness the generative capabilities of LLMs to produce targeted and contextually appropriate outputs, even in complex scenarios.

In-Context Learning: One of the key advantages of decoder-only models is their ability to perform in-context learning through zero-shot and few-shot methods.

Zero-Shot Learning: In zero-shot learning, the model is provided with a prompt that describes the task without any specific examples. The model leverages its pre-trained knowledge to generate responses that address the task based on the given instructions. This approach demonstrates the model's capability to generalize across various tasks and provide relevant outputs without additional training or examples.

Few-Shot Learning: Few-shot learning involves including a small number of examples within the prompt to illustrate how a task should be performed. These examples help the model understand the desired output format and task requirements, enhancing its performance on specific tasks. By providing a few representative instances, the model can adapt its responses more accurately to align with the given examples.

By leveraging the strengths of both encoder-only and decoder-only architectures, LLMs offer versatile solutions to a wide range of NLP challenges, from understanding and classifying text to generating sophisticated language outputs. These models continue to play a critical role in advancing the capabilities of modern NLP systems.

2.9 Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) is a powerful approach proposed by Lewis et al. [13], which enhances a language model's (LLM) capabilities by integrating external knowledge sources, such as databases or document collections. This method enables the LLM to access information that is not inherently encoded within its model parameters, thereby broadening its knowledge base and improving response quality.

The core idea behind RAG is to combine the retrieval of relevant information with generative capabilities, allowing the model to produce more informed, contextually accurate, and factually grounded outputs. By leveraging both parametric memory (the model's internal weights) and non-parametric memory (external documents), RAG has proven particularly effective in knowledge-intensive tasks, outperforming models that rely solely on internal memory. This process involves several key components, as illustrated in Figure 2.2.

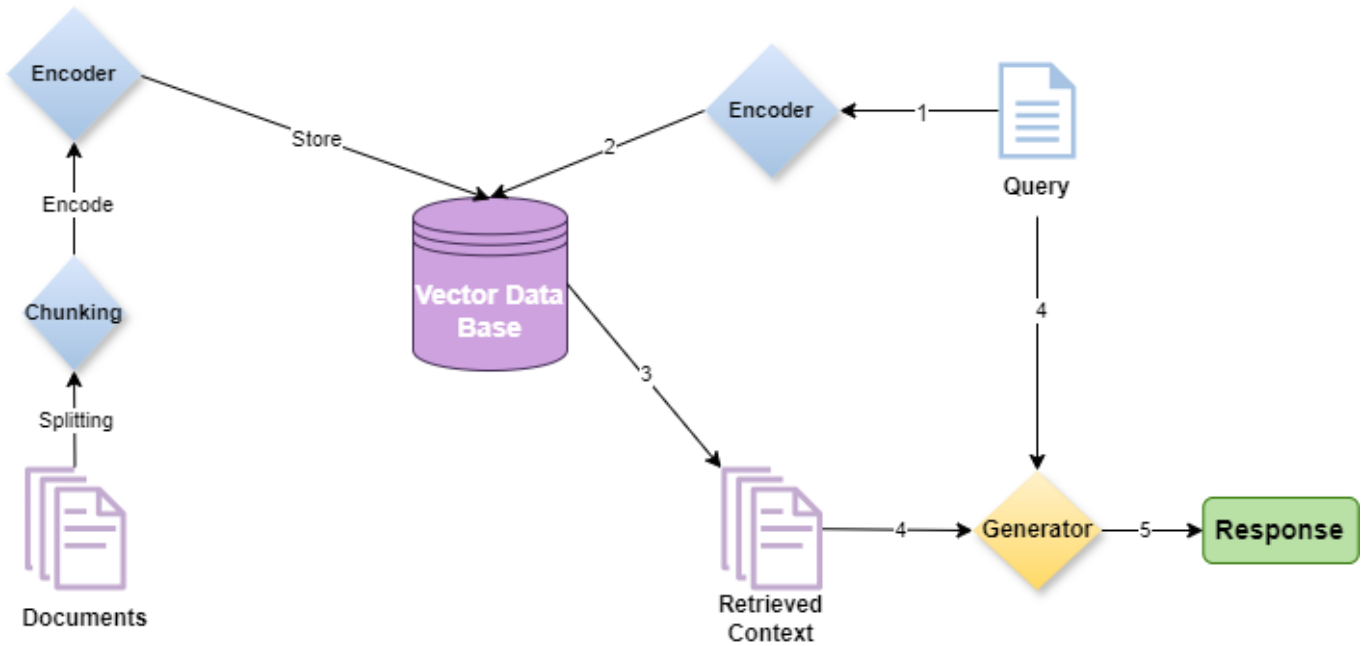


Figure 2.2: Components of the Retrieval-Augmented Generation (RAG) system.

Vector Database Creation: The process begins with the conversion of a dataset into vector representations, which are stored in a vector database. These vectors can be generated using dense embeddings from models like BERT, or through traditional sparse embeddings such as TF-IDF[20]. The choice of embedding method depends on whether semantic retrieval or term frequency-based retrieval is preferred. This step is crucial for preparing the data for effective retrieval.

User Query: When a user provides a query in natural language, this input is used to seek specific information or generate a completion. The user query is also converted into a vector representation to match the format used in the vector database. This conversion ensures that the query can be compared to stored vectors for relevance.

Information Retrieval: The retrieval mechanism then scans the vector database to find segments of text that are semantically similar to the user’s query. This involves comparing the query vector to stored vectors to identify relevant segments. These segments provide additional context to the LLM, enhancing its ability to generate a more informed and accurate response.

Combining Data: The retrieved data chunks are combined with the user’s original query to form an expanded prompt. This combined input includes both the query and the additional context from the retrieved documents. This enriched prompt provides the LLM with a broader context for generating responses.

Generating Text: Finally, the LLM processes the enlarged prompt, which now contains the additional context, to produce a context-aware response. The integration of retrieved information allows the LLM to generate responses that are not only coherent but also enriched with relevant details. In summary, RAG effectively combines retrieval and generation techniques to enhance the performance of language models by augmenting their knowledge with relevant external information.

2.10 LLM Alignment

Large Language Models (LLMs) were initially developed to predict subsequent text tokens with high accuracy. While this capability is notable, it does not inherently guarantee that the model’s outputs will be useful, safe, or aligned with human values. This creates a significant risk of generating content that may not adhere to ethical or safety standards. Consequently, aligning LLMs with human feedback has become a critical focus in their development to ensure outputs meet these standards.

Challenges in Domain-Specific Contexts: LLMs frequently encounter difficulties in producing responses that are both contextually appropriate and aligned with the specific requirements of various domains. Without targeted alignment, these models may generate outputs that are irrelevant, inaccurate, or even unethical, which could result in misuse or the spread of misinformation.

Alignment Techniques: To enhance the quality and reliability of LLM outputs, several alignment techniques have been developed:

- **Supervised Fine-tuning:** This process involves fine-tuning LLMs using high-quality, dialogue-specific datasets. The model is trained on curated prompt-response pairs to adapt its responses to specific tasks or applications. Models such as Dolly-v2 and Falcon-Instruct undergo this process to improve their conversational performance. By minimizing cross-entropy loss during this phase, the model is optimized for better understanding and generating appropriate dialogue.
- **Reinforcement Learning from Human Feedback (RLHF):** The final refinement of the model occurs through Reinforcement Learning [18], where the model is trained to maximize the scores assigned by the Reward model. This technique adjusts the model’s behavior based on extensive human feedback, allowing it to generate more contextually appropriate and higher-quality responses. Notable examples of models that use RLHF include InstructGPT and ChatGPT, which have been fine-tuned to align more closely with human preferences and ethical guidelines.

Through these alignment techniques, the limitations of LLMs can be addressed, ensuring their outputs are not only more useful and contextually relevant but also safe and ethically aligned. The alignment process plays a crucial role in improving the practical applications of these models, reducing issues such as hallucinations and misalignment with human values.

Here's a concise conclusion for the chapter:

Conclusion

In summary, this chapter provided a comprehensive overview of Large Language Models, their core architectures, and key variants. We explored how these models are pretrained and evaluated, as well as practical approaches for leveraging them, such as RAG. Understanding these concepts is essential for effectively utilizing LLMs in our project.

Chapter 3

Literature Review

3.1 Introduction

This chapter reviews the existing research on the person-job fit problem, starting with an analysis of manual resume screening processes. It examines the structure and content of resumes and job descriptions, standard HR practices for evaluating candidates, and the challenges associated with manual screening. Following this, the chapter explores automated approaches to the person-job fit problem, including the evolution and definition of Applicant Tracking Systems (ATS), the various methods employed to automate resume evaluation, a state-of-the-art overview of current solutions, and their limitations.

3.2 Manual Resume Screening

Manual resume screening is typically done by HR professionals and requires a deep understanding of both the CV and the job description, as well as their relevance to each other. It demands a certain level of expertise to accurately assess the fit.

3.2.1 Structure and Content of Resumes and Job Descriptions

Resumes and job descriptions serve as the primary tools in the recruitment process, each containing critical information that facilitates the matching of candidates to specific roles. Understanding the structure and content of these documents is essential for effective resume screening.

Structure and Content of Resumes

A resume is a structured document that provides an overview of a candidate's qualifications, experiences, and skills. While formats may vary, most resumes include the following key sections[26] :

- **Personal Information:** Typically includes the candidate's name, contact details, and sometimes a professional summary or objective statement. This section is essential for establishing the candidate's identity and providing a brief overview of their career goals.
- **Professional Experience:** Outlines the candidate's work history, usually in reverse chronological order, detailing job titles, company names, employment dates, and key responsibilities or achievements. This is the most critical part of a resume, as it demonstrates the candidate's relevant experience and expertise.

- **Education:** Lists the candidate's academic qualifications, including degrees obtained, institutions attended, and graduation dates. This section is particularly important for roles requiring specific educational credentials or certifications.
- **Skills:** Highlights the candidate's technical and soft skills relevant to the job. This section is crucial for matching the candidate's capabilities with the job's requirements.
- **Certifications and Awards:** Includes any additional certifications, licenses, or honors the candidate has received. This section can be a differentiator, particularly in competitive fields where specialized qualifications are valued.
- **Projects:** Some resumes include a section detailing specific projects the candidate has worked on, especially in technical fields. This provides insight into the candidate's hands-on experience and problem-solving abilities.

Each of these sections plays a vital role in presenting the candidate's qualifications in a clear and organized manner. The **professional experience** and **skills** sections are often the focal points during the screening process, as they directly relate to the job's requirements.

Structure and Content of Job Descriptions

A job description is a formal document that outlines the duties, responsibilities, and qualifications required for a specific role. It typically includes the following sections[25] :

- **Job Title:** Provides a clear indication of the role within the organization, setting the expectation for the level of responsibility and the nature of the work.
- **Summary of Role:** A brief overview of the position's primary purpose and key responsibilities, helping candidates quickly understand what the role entails and whether it aligns with their career goals.
- **Key Responsibilities:** Details the specific duties the employee will be expected to perform. This section is crucial for assessing whether a candidate's experience aligns with the role's requirements.
- **Required Qualifications:** Lists the minimum education, experience, and skills necessary for the position. This is a critical section for screening candidates, as it sets the baseline for eligibility.
- **Preferred Qualifications:** Outlines additional skills or experiences that are not essential but would enhance the candidate's suitability for the role. This section is important for identifying candidates who may exceed the basic requirements.
- **Company Information:** Provides background on the organization, including its mission, culture, and values. This can help candidates assess whether they would be a good fit for the company.

The job description serves as the blueprint for the recruitment process. By clearly defining the role's expectations and qualifications, it guides both recruiters and candidates in determining the suitability of a match. The **required qualifications** and **key responsibilities** sections are particularly

important during resume screening, as they allow for the direct comparison of candidate credentials with job needs.

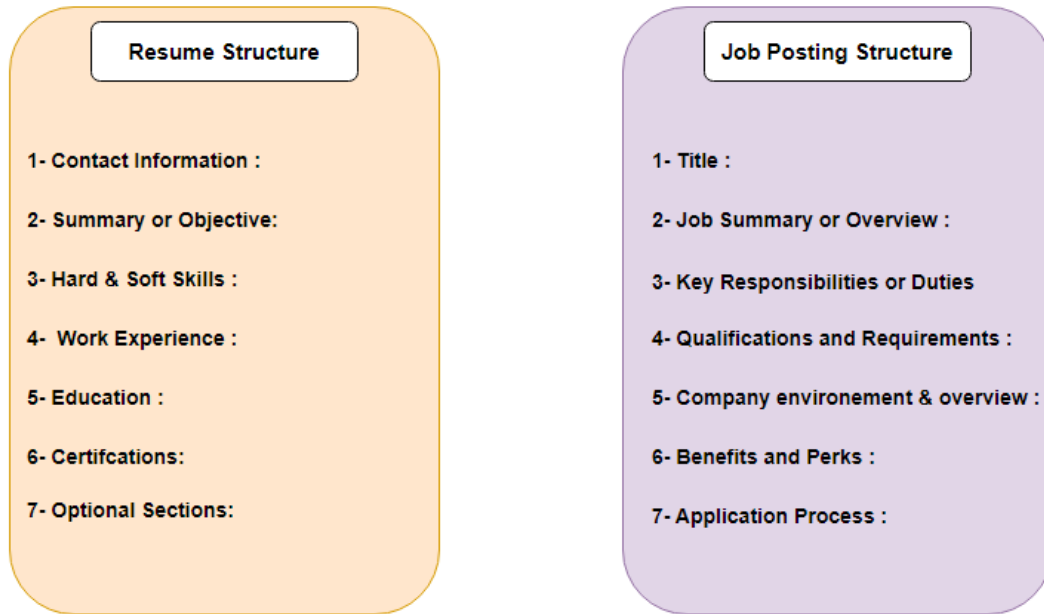


Figure 3.1: Overview of the Structure of Resumes and Job Descriptions.

Importance in Resume Screening

Understanding the structure and content of resumes and job descriptions is essential for effective resume screening. Each section of these documents provides specific information that can be used to assess a candidate's fit for a role.

The alignment between the **professional experience** and **skills** sections of a resume with the **key responsibilities** and **required qualifications** in a job description is often the primary focus during the screening process. By meticulously analyzing these sections, recruiters can efficiently identify the most suitable candidates, thereby optimizing the recruitment process.

3.2.2 Resume Screening Techniques for HR

Resume screening is a crucial step in narrowing down candidates based on qualifications and skills. Typically conducted by HR professionals, this manual process allows for a more nuanced and thorough evaluation. HR professionals assess key criteria such as relevant experience, skills, and qualifications to ensure candidates are a good match for the role [24].

Steps in Resume Screening:

1. **Check Required Credentials:** Ensure candidates meet the basic job requirements, such as specific experience or certifications. Resumes lacking these criteria can be excluded early in the process.

2. **Evaluate Desired Skills:** Look for candidates who possess both required and preferred skills, including relevant technical abilities and soft skills like leadership or communication, which may enhance team integration.
3. **Look for Customization:** Customized resumes that align with the job posting indicate attention to detail and research, reflecting genuine interest in the role.
4. **Verify Information:** Conduct reference checks to confirm the accuracy of credentials and assess qualities like integrity and teamwork.

By applying these screening techniques, recruiters can effectively identify qualified candidates while ensuring the best cultural fit for the organization.

3.2.3 Challenges and Limitations of Manual Screening

Manual resume screening presents several significant challenges that affect both its effectiveness and efficiency.

One of the primary issues is the sheer volume of applications that recruiters must process. On average, recruiters receive around 250 resumes for each job opening, with approximately 86-90% of these applications being irrelevant. This means that a substantial amount of time is spent reviewing resumes that do not meet the job criteria. The manual nature of the screening process often leads to a first-come, first-served approach, resulting in time wasted on unsuitable candidates. This inefficiency can create the impression that finding qualified candidates is difficult, potentially leading to compromises in the quality of hires and settling for less suitable candidates.

Another challenge is the domain expertise required for effective resume screening. Recruiters frequently handle applications across various fields, such as marketing, sales, IT, and engineering. It is unrealistic to expect recruiters to possess in-depth knowledge of every specialized area. This lack of specialized expertise can result in the inappropriate shortlisting of candidates, as recruiters may not fully grasp the specific requirements of different roles. As a result, hiring managers often need to take on a significant portion of the screening process, which can be an inefficient use of their time and resources.

Additionally, the limitations of current recruitment technologies further exacerbate these challenges. Applicant Tracking Systems (ATS) is commonly used tool in the recruitment process. ATS can streamline certain aspects by filtering candidates based on binary criteria such as qualifications, experience, and location. However, ATS systems often rely on keyword matching, which can be problematic. This method may favor resumes with a high density of keywords and overlook well-qualified candidates whose resumes are less keyword-rich. These limitations highlight the need for more advanced and nuanced approaches to resume screening.

Addressing these challenges is crucial for improving the efficiency and accuracy of the resume screening process. Advanced methods and technologies are needed to enhance the effectiveness of candidate selection and reduce the reliance on manual and often inefficient screening practices.

3.3 Related Work on the Person-Job Fit Problem

In this section, we will delve into the evolution of recruitment technologies and their impact on person-job fit, examining the development of Applicant Tracking Systems (ATS), various approaches to person-job fit, and the current state of the art in this field.

3.3.1 The Evolution and Definition of Applicant Tracking Systems

Over the years, recruitment has seen a huge shift due to the need for faster and more effective hiring methods. With the growth of the internet and new software, there was a clear demand for automation in recruitment. This led to the development of Applicant Tracking Systems (ATS), specialized tools designed to simplify the hiring process by automating many tasks that used to be done manually by HR teams.

Early ATS: The 1960s and 1980s

ATS first appeared in the 1960s when IBM created basic software to store resumes and perform simple keyword searches. While useful, these early systems had limited functionality. In the 1980s, more advanced ATS started to emerge, but they were still focused on simple tasks like organizing resumes without the ability to deeply analyze them.

Although these systems helped manage candidate information, they lacked advanced features like resume parsing or detailed analytics.

Modern ATS: The 2000s and Beyond

ATS technology made significant advancements in the early 2000s. Modern systems could now break down resumes into sections like contact details, work experience, and skills, making it easier for HR teams to review and manage candidates. ATS also started integrating with other HR tools, further streamlining the hiring process.

Advanced ATS with AI Integration

In recent years, ATS has incorporated AI technologies like machine learning and Natural Language Processing (NLP). These tools allow the system to better understand and analyze resumes, moving beyond simple keyword matching to provide more accurate candidate recommendations, which improves hiring quality and speeds up the process.

The introduction of AI has been a game-changer for ATS, enabling systems to evolve and continuously improve. This integration has transformed how companies recruit, setting the stage for even more advances in HR technology.

In the next section, we will explore how these systems work and their role in improving recruitment processes.

3.3.2 Approaches to the Person-Job Fit Problem

The person-job fit problem has evolved significantly with advancements in technology. Initially addressed through basic rule-based methods, the approach has transitioned to more sophisticated machine learning techniques and, more recently, to advanced deep learning and large language models

(LLMs). This section reviews the evolution of these approaches and explores contemporary methods in the context of state-of-the-art LLM solutions.

Historical Evolution of Approaches

Early approaches to the person-job fit problem were largely rule-based. These systems relied on pattern matching to identify relevant keywords and compare them against job descriptions. Such methods were straightforward, computing the percentage of matching keywords between resumes and job requirements. Rule-based systems also used heuristics to categorize information into predefined sections, such as education and experience.

The advent of machine learning brought a significant shift in the approach to person-job fit. Machine learning algorithms, capable of analyzing large datasets of resumes and hiring outcomes, began to offer improved accuracy and efficiency in screening processes. These algorithms learn patterns and correlations from historical data, enhancing the ability to match candidates to job profiles.

General Approaches with Large Language Models

Recent advancements in large language models (LLMs) have introduced new possibilities for addressing the person-job fit problem. These models, such as BERT and GPT-3, excel in natural language processing and understanding, offering improved methods for text representation. The state-of-the-art approaches include:

- **Text-Based Matching:** Leveraging LLMs to generate more nuanced document representations for both resumes and job descriptions. This approach enhances semantic similarity matching by understanding the context and content of textual data, thus improving the relevance of job-candidate matches.
- **Recommendation Systems:** Casting the problem as a recommendation task, LLMs can utilize collaborative filtering methods, such as user-based and item-based recommendations, to suggest suitable jobs for candidates and vice versa. This approach is particularly effective in recruitment platforms where interaction data is available, though it is less applicable in enterprise settings due to limited interaction data.

The integration of LLMs into recruitment processes represents a significant advancement, allowing for more sophisticated and context-aware matching techniques. However, challenges remain in applying general-domain LLMs to domain-specific data, as domain-specific structures and semantics may not always align with pretrained models.

As the field continues to evolve, it is crucial to explore these state-of-the-art methods in greater detail. The next section will delve into a review of key papers and recent advancements in this area, highlighting the impact of these technologies on improving person-job fit and recruitment outcomes.

3.3.3 State of the Art Overview

Building upon the discussion in the previous subsection on various approaches to the Person-Job Fit problem, this section provides an overview of the state-of-the-art methodologies. Table 3.1 presents a comparison of key papers and algorithms that represent the forefront of research in this domain. Each entry in the table summarizes the objectives these approaches aim to achieve, the

core methods employed, their notable strengths, and any limitations identified. Additionally, the table highlights the improvements these approaches have demonstrated over baseline methods. This comprehensive overview offers a clear picture of the current advancements and the challenges that still remain in achieving an optimal person-job fit.

Table 3.1: Comparison of Key Approaches in Person-Job Fit Research

Title	Objective	Method	Strengths	Limitations	Results
Towards Effective and Interpretable Person-Job Fitting [12]	Ranking	Multi-task optimization, deep learning	High recommendation accuracy, interpretability	Does not fully consider psychological motivations	Outperforms baselines
Resume Shortlisting and Ranking with Transformers [10]	Ranking	Sentence-BERT (SBERT)	Efficient shortlisting, better coherence	Does not account for personalized factors and soft skills	SBERT shows better performance than BERT
Learning to Match Jobs with Resumes from Sparse Interaction Data using Multi-View Co-Teaching Network (MV-CoT) [2]	Classification	Multi-view co-teaching, text and relation-based models	Handles sparse, noisy data	Requires sufficient labeled data	Outperforms in sparse and noisy data handling
Generative Job Recommendations with Large Language Model (GIRL) [30]	Job Recommendation	LLM, supervised fine-tuning, reinforcement learning	Personalized job suggestions, no candidate set	Lack of transparency in JD generation	Improves personalized job recommendations
TAROT: A Hierarchical Framework with Multitask Co-Pretraining on Semi-Structured Data towards Effective Person-Job Fit [3]	Ranking, Classification	Hierarchical multitask co-pretraining	Better embeddings, significant performance	May not generalize well to different datasets	Significant improvements over baselines

Conclusion: The table provides a comparative analysis of advanced approaches in the person-job fit domain. Each method offers unique strengths, such as improved interpretability, better handling of noisy data, or enhanced recommendation capabilities. However, limitations such as lack of consideration for psychological factors or high computational demands persist. Overall, these advancements represent significant strides towards optimizing person-job fit, yet ongoing research is needed to address remaining challenges and further refine these methodologies.

3.3.4 Limitations of State-of-the-Art Approaches

Despite the significant advancements in methodologies for the Person-Job Fit problem, several limitations persist in the state-of-the-art approaches. Key limitations include:

- **Generalization Issues:**
 - Many approaches, particularly those leveraging deep learning and complex embeddings, struggle with generalization across different datasets and domains.
 - Performance can degrade when these models are applied to new or diverse job markets, limiting their robustness in real-world scenarios.
- **Data Sparsity and Quality:**
 - Techniques like multi-view co-teaching networks and reinforcement learning require large amounts of high-quality labeled data.
 - In practice, data can be sparse or noisy, reducing the effectiveness of these models and leading to suboptimal recommendations.
- **Computational Complexity and Resource Requirements:**
 - Many state-of-the-art methods involve complex algorithms and large models that demand significant computational resources.
 - This high computational cost can be a barrier to deployment in resource-constrained environments, making such methods less accessible for smaller organizations or in low-resource settings.

3.4 Conclusion

In this chapter, we have reviewed the evolution of resume screening from manual methods to modern ATS with AI integration. We highlighted advancements in addressing person-job fit, while also noting ongoing limitations. The insights provided underscore the need for continued innovation in recruitment technologies to improve hiring accuracy and efficiency. The next step is to begin working on the project to apply these insights and develop enhanced solutions.

Chapter 4

Data Collection and Preparation Pipeline

4.1 Introduction

This chapter addresses the essential processes of data collection and preparation for developing our resume screening solution. It encompasses strategies for acquiring diverse datasets, including job descriptions and resumes from multiple sources, and outlines the preprocessing steps necessary to standardize and enhance data quality. Additionally, it introduces the evaluation protocol for assessing the accuracy and relevance of the solution, providing a clear framework for building and refining the resume matching model.

4.2 Data Collection and Preparation Pipeline

The data collection and preparation pipeline serves as the foundational framework for gathering and processing diverse data relevant to resumes and job descriptions. This pipeline is structured into several stages, each playing a crucial role in ensuring the integrity and usability of the data.

The initial stage involves the aggregation of various types of related data from multiple sources into a centralized data lake. This data lake, implemented as an AWS S3 bucket, accommodates a wide array of data formats, including resumes and job postings. The aim is to consolidate disparate datasets to facilitate further processing.

Subsequently, the collected data undergoes a processing stage designed to standardize and translate the content into English. This standardization ensures consistency across the dataset, allowing for more effective analysis in later stages.

Following the processing stage, an inference pipeline is employed to extract relevant metadata, providing additional context and structure to the data. This is followed by an exploratory data analysis (EDA) stage, where insights are gleaned from the data, revealing patterns and trends that inform subsequent actions.

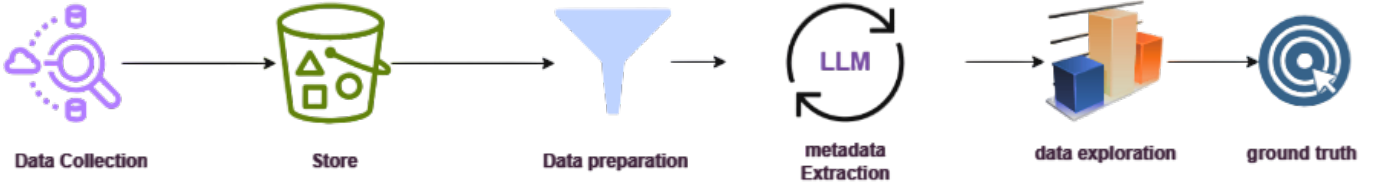


Figure 4.1: Overview of the Data Collection and Preparation Pipeline

Finally, the processed data is transitioned to the ground truth preparation stage, where it is curated and organized to serve as a reliable benchmark for evaluation purposes.

The subsequent sections will delve into each of these components in detail, outlining the methodologies and tools employed throughout the pipeline.

4.3 Data Collection

Since datasets that link job postings to matching CVs are not readily available, we need to gather these datasets separately. Once acquired, the data can be labeled manually to pair job postings with the most relevant CVs. This step is critical for building and testing the system’s ability to accurately match candidates to job descriptions.

4.3.1 Resume Datasets

- **Resume Corpus [7]:**
 - A multi-labeled dataset of resumes categorized by occupation.
 - Includes:
 - * *resumes_corpus.zip*: Individual resume files in `.txt` format, with corresponding labels in `.lab` format.
 - * *resumes_sample.zip*: A consolidated text file where each line represents a resume, containing the resume ID, list of occupations, and the resume text itself, separated by `":::"`.
 - * *normalized_classes*: A file mapping occupation names to their normalized forms.
- **Curriculum Vitae Data [16]:**
 - Contains 3,266 resumes in both PDF and DOC formats, sourced from India.
 - Covers a wide range of job categories, providing a diverse set of examples.
 - A related repository offers a K-means clustering algorithm for categorization tasks.
- **Resume Dataset [1]:**
 - Comprises over 2,400 resumes in both string and PDF formats.
 - Resumes are categorized into 25 job-related categories, including IT, Healthcare, and Engineering.

- PDF files are organized by category, and a CSV file maps resume IDs to their respective categories.
- **Chinese Resume Corpus [22]:**
 - A large collection of 178,000 Chinese resumes, containing over 33 million words.
 - Unstructured resumes include various types of common information such as name, gender, birthday, education, and work experience.
 - Evaluated using four types of mainstream neural network models, demonstrating its potential for information extraction research.
- **Resume Text Classification Dataset [11]:**
 - Created by converting various resume formats into a uniform text format.
 - Includes a pilot study that annotated 500 out of 15,000 original CVs from Kaggle.
 - Resumes are labeled with five categories: experience, knowledge, education, project, and others.
- **Job Postings Dataset from Hugging Face [23]:**
 - Contains a large collection of job postings across multiple domains.
 - Available in various formats, including CSV, PDF, and TXT.
 - The dataset comprises approximately 29,000 covering multiple domains.

4.3.2 Metadata Extraction

In order to maximize the utility of our dataset, we implemented a multi-step process to extract and organize metadata from both resumes and job postings. Given the diversity and unstructured nature of the data, we developed a comprehensive approach to structure it effectively for subsequent analysis.

The first challenge we encountered involved integrating non-English resumes into the dataset. Specifically, we translated a set of high-quality Chinese resumes, which were sourced from the research corpus detailed in [22] containing 178,000 documents. This translation, performed via the Google Translate API, was critical in ensuring uniformity in metadata extraction. Translating these resumes allowed us to maintain consistency across diverse data sources, facilitating more accurate and reliable metadata extraction.

Following the translation, we employed a Large Language Model (LLM) to infer metadata from the resumes and job postings. The LLM was guided by carefully crafted prompts to focus on extracting specific information. For resumes, the LLM categorized candidates into three experience levels:

- **Entry-Level:** Focusing on education, internships, and fundamental skills.
- **Mid-Level:** Highlighting professional experience, achievements, and specialized skills.
- **Senior-Level:** Identifying leadership experience, strategic impact, and advanced expertise.

For job postings, the model classified documents based on job roles and industries.

The success of the metadata extraction process was highly dependent on the quality of the prompts. Clear and detailed prompts were provided to ensure the LLM extracted the desired fields and reduced hallucinations, where the model might generate irrelevant or incorrect outputs. The LLM was instructed to output metadata in a structured format, guiding it to follow instructions strictly and reason logically during the extraction process.

To assess the effectiveness of this approach, we validated the metadata extraction on a test dataset of 3,000 AI-related resumes. The LLM achieved an accuracy rate of 90%, as illustrated in the confusion matrix in Figure 4.2. This high level of accuracy demonstrates the reliability of the LLM in extracting metadata from both resumes and job postings.

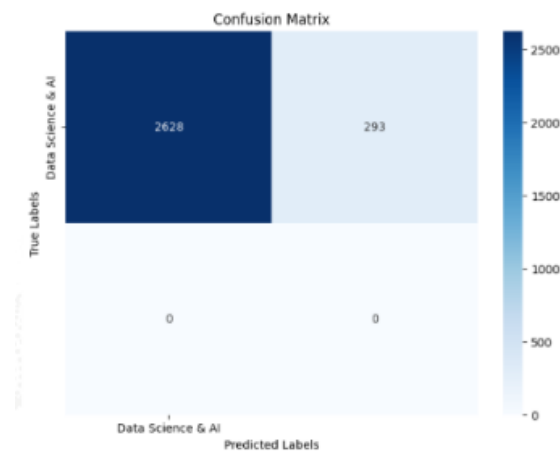


Figure 4.2: Confusion Matrix for the labeling of AI jobs.

By leveraging effective prompting techniques, we successfully extracted and organized metadata from large and diverse datasets, ensuring consistency, accuracy, and scalability across both resumes and job postings.

4.3.3 Visualization and Exploratory Analysis

Following the extraction of metadata, we conducted analysis to explore the dataset’s characteristics, which encompass job categories and experience levels. The visualizations presented below provide insights into the distribution of job postings and resumes across different categories and experience levels.

The dataset includes nine categories: IT, AI, Business, HR, Sales, Design, Engineering, Training & Education and Others. The distribution of job postings is illustrated in Figure 4.3. As shown, AI Data Science represents 27.3% of the postings, indicating a strong demand in this sector. Engineering follows with 15.1%, while Business and HR Administration constitute 9.2% and 10.9%, respectively. Other categories collectively account for 33.4% of the dataset. This distribution highlights the predominant focus on technology and data science roles, with a substantial presence in engineering and administrative positions.

Figure ?? depicts the distribution of resumes across various categories. IT resumes dominate with 43%, reflecting the high volume of candidates in this field. Engineering resumes make up 25.4%, while

HR Administration contribute 12.7%. Other categories account for 1.9%, suggesting a concentration of resumes in the technology sector. The high proportion of IT-related resumes aligns with the demand for technology roles observed in job postings.

The distribution of job postings by experience level is shown in Figure 4.5. Mid-level positions are most prevalent, comprising over 6,000 postings, which signifies a significant demand for professionals with substantial experience. Senior-level positions follow with over 2,500 postings, while entry-level and executive roles are less common, with approximately 1,000 and fewer than 500 postings, respectively. This distribution indicates a robust market for mid-career professionals and fewer opportunities for entry-level and executive positions.

Figure 4.6 illustrates the distribution of resumes by experience level. Senior-level resumes are the most numerous, exceeding 5,000, followed by mid-level resumes at around 4,000. Entry-level and executive resumes are relatively scarce, each with fewer than 500 entries. This pattern suggests that the supply of senior-level candidates is strong, which could be attributed to the higher experience and expertise of these individuals.

Overall, the visualizations reveal key trends in the job market and candidate profiles. The high volume of IT and engineering job postings and resumes underscores the emphasis on these fields. The prevalence of mid- and senior-level positions in job postings and resumes reflects a substantial focus on experienced professionals. Despite the partial labeling of the full dataset, these visualizations provide valuable insights for further analysis and model training.

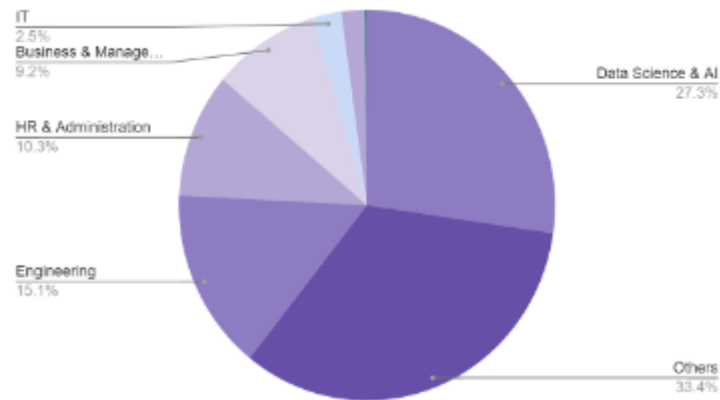


Figure 4.3: Distribution of job postings across various categories.

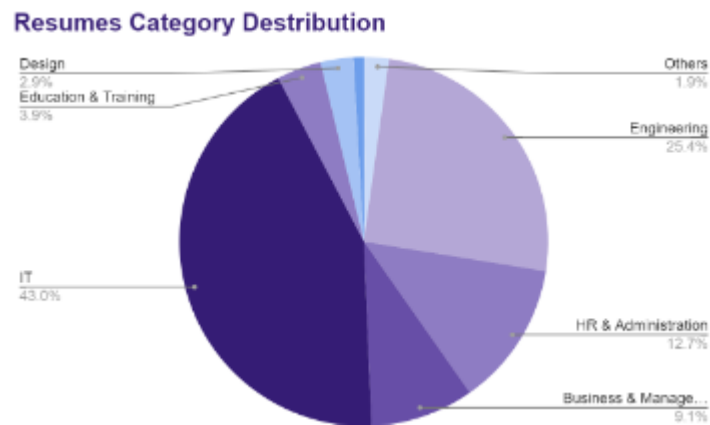


Figure 4.4: Distribution of resumes across different categories.

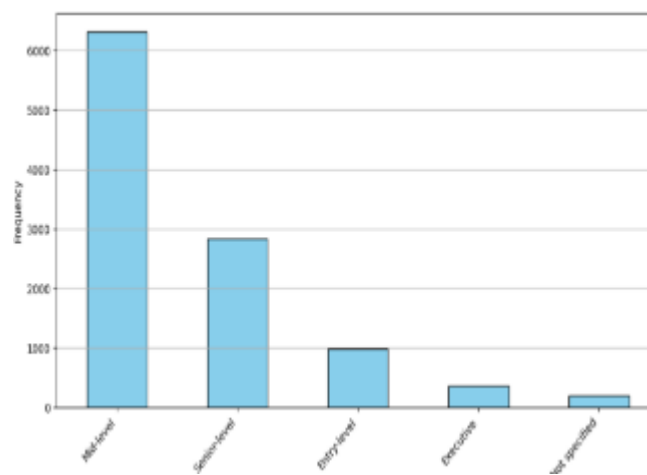


Figure 4.5: Distribution of job postings by experience level

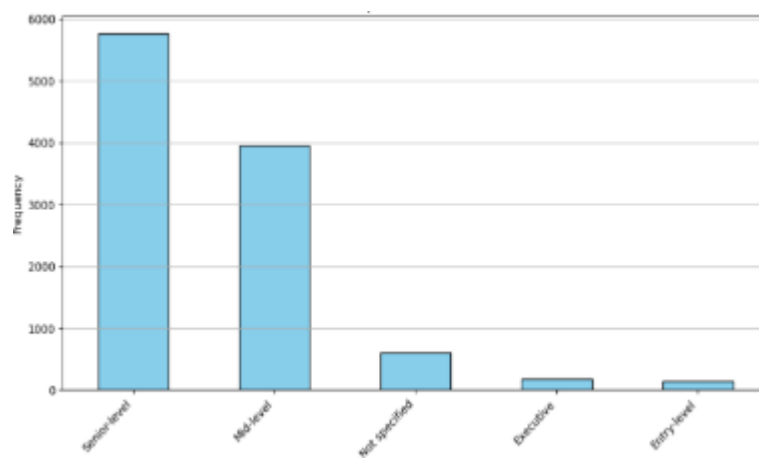


Figure 4.6: Distribution of resumes by experience level.

In conclusion, the metadata extraction process has resulted in a well-structured large dataset that integrates job postings and resumes with categorized metadata. This dataset is invaluable for research, model training, and generating insights into job market trends and candidate qualifications. The comprehensive and structured approach ensures that the data is suitable for advanced analysis and application in various machine learning models.

4.4 Creation of the Ground Truth Dataset

In this section, we present a detailed and methodologically rigorous approach to constructing our ground truth dataset, which plays a crucial role in validating the performance of our ranking system. Given the sheer scale of over 230,000 collected documents, it was impractical to evaluate each CV for every job description. Therefore, a systematic and semi-automated process was developed to create a representative and reliable dataset, ensuring that our results are both generalizable and valid across various job categories and experience levels.

The process begins with the selection of a random subset of 30 CVs for each job description. This subset is not chosen arbitrarily; rather, it is filtered based on metadata that ensures the CVs belong to the same job category and experience level as the job description. This step is critical in focusing our evaluation on candidates who are more likely to be relevant to the job, thereby enhancing the quality of the labeled data.

Once the subset is established, we apply a carefully designed screening protocol, based on best practices from industry guidelines, to score and rank the candidates. The protocol is outlined as follows:

Identification of Job Requirements: For each job description, we meticulously identify key requirements, dividing them into two categories:

- *Minimum Requirements:* These are essential qualifications, skills, certifications, and experiences that a candidate must possess to be considered for the role.
- *Additional Requirements:* These are desirable but not mandatory qualifications, skills, certifications, and experiences that could further enhance a candidate's suitability for the role.

Scoring of Candidates: Each candidate in the subset is evaluated against the identified requirements. The scoring system is designed to reflect how well each candidate meets these requirements:

- A score of 0 is assigned if the requirement is not met.
- A score of 0.5 is assigned if the requirement is partially met.
- A score of 1 is assigned if the requirement is fully met.

Computation of Total Scores: We compute the total scores by summing the scores for minimum requirements and additional requirements separately. The overall score for each candidate is then calculated using the following formula:

$$\text{Overall Score} = 2 \times \text{Minimum Requirements Score} + \text{Additional Requirements Score}$$

This weighted formula ensures that meeting the minimum requirements is prioritized, reflecting their critical importance.

Classification and Ranking of Candidates: Candidates are classified into two groups based on their minimum requirements score:

- *Top Candidates:* Those with a minimum requirements score exceeding 80%.
- *Worst Candidates:* Those with a minimum requirements score of 80% or less.

Within each group, candidates are further ranked based on their overall score.

Handling of Edge Cases: To ensure fairness and accuracy, we account for potential edge cases in our scoring:

- *Overqualified Candidates:* Candidates who are deemed overqualified for the role are directly placed in the worst candidate group, regardless of their score, to prevent misalignment between candidate qualifications and job requirements.
- *Contextual Relevance of Experience:* Candidates whose experience, though extensive, is less relevant to the job are penalized by 2 points in their overall score.
- *Gaps in Employment:* Employment gaps are penalized by 2 points to account for potential concerns regarding continuity or commitment.

Final Dataset Construction: This rigorous process allowed us to generate a ground truth dataset that includes 30 job descriptions, each paired with its top 5 and worst 5 ranked candidates. This required an exhaustive review of over 1,000 CVs, ensuring that our dataset is both comprehensive and diverse. To enhance the generalizability of our ground truth, the 30 selected CVs for each job were drawn from different job categories and experience levels. This approach ensures that our results are not biased toward any specific job type or experience level, thereby supporting the broader applicability of our findings.

In cases where the initial subset of 30 CVs did not yield a sufficient number of top matches, we expanded the subset by an additional 30 CVs to ensure a robust ranking process. This additional step demonstrates our commitment to thoroughness and precision in the creation of the dataset.

Given the complexity and scale of this task, we developed a semi-automated console application to assist in the labeling process. This tool facilitated the manual filtering of related CVs, scoring candidates based on their match or non-match to the job description, and ranking them accordingly. The use of this application was instrumental in managing the substantial workload and ensuring consistency across the dataset.

In conclusion, the creation of this ground truth dataset represents a significant achievement in our research. By rigorously selecting, scoring, and ranking candidates from a diverse range of job categories, we have established a reliable and generalizable dataset that will serve as a foundation for evaluating our ranking system.

4.5 Evaluation Protocol

The task of ranking and shortlisting top candidates for job postings requires not only retrieving the most relevant candidates but also ensuring that they are ranked correctly. Given that hiring decisions often focus on a small set of top candidates, it is essential to develop an evaluation framework that effectively assesses both candidate retrieval and ranking.

Standard evaluation metrics, such as *Precision* and *Recall*, provide strong indicators of how well the system retrieves the most relevant candidates from a large pool. However, these metrics alone are insufficient when the ranking of candidates plays a critical role. In the recruitment context, even if the system retrieves the correct set of candidates, presenting them in an incorrect order may lead to suboptimal decision-making, with more qualified candidates appearing lower in the list.

Therefore, a robust evaluation protocol is required, one that accounts for both the retrieval of relevant candidates and their correct ranking. To address this, we propose a two-stage evaluation protocol. The first stage assesses the *retrieval performance*, while the second stage evaluates the *ranking quality* of the retrieved candidates. This dual approach ensures that the system is both capable of identifying the most suitable candidates and ranking them appropriately, which is critical for successful shortlisting in real-world recruitment scenarios.

4.5.1 Stage 1: Evaluating Retrieval Performance

The first stage of the evaluation focuses on measuring how well the system retrieves relevant candidates from the pool of all candidates. The objective is to ensure that the system correctly identifies the candidates to be considered further. The key metrics for evaluating retrieval performance are:

- **Precision@k**: Measures the proportion of relevant candidates among the top- k retrieved results.

$$\text{Precision@k} = \frac{\text{Number of relevant candidates in top-}k}{k} \quad (4.1)$$

This metric indicates the quality of the retrieved candidates at various cutoff points (e.g., top 5 or top 10 candidates).

- **Recall@k**: Measures the proportion of relevant candidates that are included in the top- k results.

$$\text{Recall@k} = \frac{\text{Number of relevant candidates in top-}k}{\text{Total number of relevant candidates in the pool}} \quad (4.2)$$

Recall@k ensures that no relevant candidates are missed in the top results.

4.5.2 Stage 2: Evaluating Ranking Quality

After retrieving relevant candidates, the system must rank them in the correct order. Poor ranking can undermine the decision-making process, even if the right candidates are retrieved. In this stage, we evaluate the system's ability to rank candidates appropriately using the following metrics:

- **Mean Reciprocal Rank (MRR)**: Measures the rank of the first relevant candidate, giving higher importance to systems that rank relevant candidates earlier.

$$\text{MRR} = \frac{1}{|Q|} \sum_{q=1}^{|Q|} \frac{1}{\text{Rank of first relevant candidate in query } q} \quad (4.3)$$

MRR is useful for understanding how quickly the first relevant candidate appears in the ranked list.

- **Average Precision (AP)**: Calculates precision at each rank where a relevant candidate is retrieved, averaged across ranks.

$$\text{AP} = \sum_{k=1}^N \frac{\text{Precision@k} \times \text{Indicator}(\text{relevant@k})}{\text{Total number of relevant candidates}} \quad (4.4)$$

AP assesses ranking precision by accounting for the positions of all relevant candidates.

- **Mean Average Precision (MAP)**: The mean of AP scores across multiple queries, providing an overall performance score for ranking.

$$\text{MAP} = \frac{1}{|Q|} \sum_{q=1}^{|Q|} \text{AP}(q) \quad (4.5)$$

MAP is the standard metric for evaluating ranking performance across multiple job postings.

- **Discounted Cumulative Gain (DCG@k)**: Evaluates ranking quality by assigning higher importance to relevant candidates appearing higher in the list. The gain is discounted logarithmically as rank increases.

$$\text{DCG@k} = \sum_{i=1}^k \frac{\text{relevance of candidate } i}{\log_2(i+1)} \quad (4.6)$$

DCG@k gives more weight to higher-ranked relevant candidates and is useful for scenarios where the ranking position matters.

- **Normalized Discounted Cumulative Gain (NDCG@k)**: Normalizes the DCG score by comparing it to the ideal ranking, ensuring a score between 0 and 1.

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{Ideal DCG@k}} \quad (4.7)$$

NDCG@k provides a robust evaluation of ranking quality by comparing the system's ranking with the optimal one.

This two-stage evaluation protocol ensures that both the *retrieval* and *ranking* performance are effectively measured, leading to a more accurate and efficient candidate shortlisting process in real-world recruitment scenarios.

4.6 Conclusion

This chapter presented a systematic approach to constructing a ground truth dataset essential for evaluating candidate-job matching systems. By defining clear screening criteria based on industry standards, we ensured an objective and reliable process for selecting and ranking candidates. Our method involved selecting 30 relevant CVs per job description and applying our screening protocol to identify the top 5 and worst 5 candidates.

Additionally, we developed an evaluation protocol tailored specifically to the task of candidate retrieval and ranking. This protocol, designed to assess both the retrieval of relevant candidates and their correct ranking, provides a robust framework for evaluating the performance of automated candidate shortlisting systems.

Chapter 5

Experiments

5.1 Introduction

In this chapter, we introduce the working environment as well as the used methodology, then we present the experimental approaches developed to build an effective ranking system for matching candidate resumes to job descriptions. The process evolved through incremental improvements, starting with basic semantic search techniques and progressing toward more advanced strategies that incorporated hybrid search methods and sophisticated reranking algorithms.

Each stage of experimentation was carefully designed to overcome specific challenges encountered in the previous phases, with the ultimate goal of improving the accuracy and relevance of the ranked resumes. This chapter outlines these developments, highlighting the insights gained, the limitations addressed, and the rationale behind each enhancement.

5.2 Working Environment

The project was executed in a well-equipped environment that combined various software tools and hardware resources:

5.2.1 Software Environment

- **Python:** The primary programming language used for data analysis and machine learning tasks.
- **Visual Studio Code (VS Code):** The Integrated Development Environment (IDE) for efficient coding and debugging.
- **GitLab:** Used for code integration and version control, facilitating collaboration and tracking changes.
- **Hugging Face:** The repository for importing various LLM models easily.
- **OpenAI:** Used for accessing non-open-source LLM models.
- **PyTorch:** The deep learning framework employed for training machine learning models.

- **Docker:** A containerization platform that allows us to package applications and their dependencies into isolated environments, ensuring consistency across different stages of development and deployment.
- **AIchor¹:** The company’s platform used to run experiments on GPU resources, triggered by code pushes to a Git repository.

5.2.2 Hardware Environment

For this project, we relied on both local and cloud-based hardware resources:

- **Local Machine:** We used a laptop with an Intel i5 processor, 24 GB of RAM, and a 225 GB SSD for our development tasks.
- **JupyterHub:** For more intensive computations, we utilized JupyterHub, which provides access to high-performance Tesla A100 and V100 GPUs. This platform also offers on-demand CPU and RAM resources, allowing us to efficiently handle large-scale machine learning tasks and experiments as needed.

5.3 Working Methodology

For this project, we utilized a hybrid approach, combining the CRISP-DM framework with Agile practices to maintain a balance between structure and flexibility.

We followed the CRISP-DM framework, starting with Business Understanding, where we defined the project’s objectives and success criteria. Next, we moved to the **Data Understanding** phase, which involved exploring and analyzing the dataset. In the **Data Preparation** phase, we focused on transforming the data and the evaluation set to ensure their readiness for the modeling stage. In the **Modeling** phase, LLM-Based systems were developed and refined to meet the project’s goals. Finally, in the **Evaluation** phase, we rigorously tested the approaches to assess their performance and ensure alignment with the established objectives.

Alongside this, we adhered to Agile principles by organizing the project into two-week **sprints**, with **daily standup meetings** to track progress. At the end of each month, we conducted a **feedback session** to evaluate overall progress and adjust the working approach if necessary.

5.4 Semantic Search Approach

To address the problem of matching and ranking candidates for job fit, we begin by employing a basic yet effective approach: **semantic search**. Semantic search is widely recognized for its application in information retrieval, where it excels in understanding and matching the contextual meaning of different documents. In this context, we aim to match candidate resumes with job descriptions based on their semantic similarity.

As discussed in the earlier chapter on large language models (LLMs), LLM embeddings possess the ability to capture the nuanced context of a given text and represent it as dense vectors. These

¹AIchor’s documentation: <https://docs.aichor.ai/>

vectors encode the semantic essence of the resumes and job descriptions in such a way that they can be compared based on meaning rather than just keyword matching.

5.4.1 System Setup

To set up our initial system, as shown in Figure 5.1, we needed two main components:

- A **vector database** to efficiently store and retrieve dense embeddings. We used **Faiss** (Facebook AI Similarity Search), a library designed for fast and scalable similarity search, as our vector database to store CVs. Faiss enables us to perform similarity searches across large datasets efficiently ².
- An **embedding model** to generate dense vector representations of both resumes and job descriptions, it should be able of encoding large documents, such as CVs and job descriptions, into dense vector representations. For this, we leveraged the **Massive Text Embedding Benchmark (MTEB)** [15]. Based on the MTEB leaderboard 5.2, we selected **SFR-Embedding-2_R** ³ as our embedding model. This model, with its 7 billion parameter decoder architecture and a hidden size of 4096, offers a maximum token limit of 32,768, allowing us to effectively handle large CVs in our processing tasks.

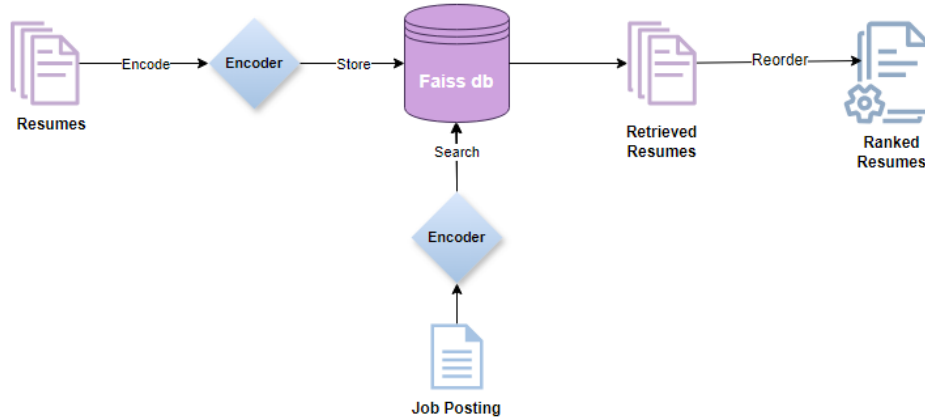
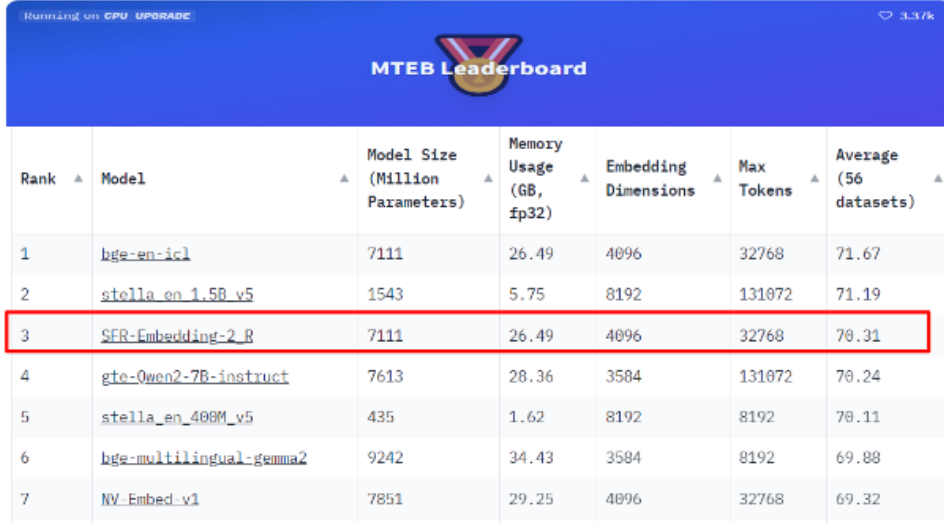


Figure 5.1: Workflow in the Semantic Search System

²<https://faiss.ai>

³model card at https://huggingface.co/Salesforce/SFR-Embedding-2_R



Rank	Model	Model Size (Million Parameters)	Memory Usage (GB, fp32)	Embedding Dimensions	Max Tokens	Average (56 datasets)
1	bge-en-v1.1	7111	26.49	4096	32768	71.67
2	stella_en_1.5B_v5	1543	5.75	8192	131072	71.19
3	SFR-Embedding-2_B	7111	26.49	4096	32768	70.31
4	gte-Open2-7B-instruct	7613	28.36	3584	131072	70.24
5	stella_en_400M_v5	435	1.62	8192	8192	70.11
6	bge-multilingual-gemma2	9242	34.43	3584	8192	69.88
7	NV-Embed-v1	7851	29.25	4096	32768	69.32

Figure 5.2: Massive Text Embedding Leaderboard.

5.4.2 Similarity Calculation

Once the resumes and job descriptions were converted into embeddings, we calculated the similarity between them using **Cosine Similarity**. The formula for cosine similarity is:

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\| \|B\|} \quad (5.1)$$

Where A and B are the embedding vectors of a resume and job description, respectively. A value of 1 indicates identical meaning.

5.4.3 Initial Results

The initial results of the semantic search-based approach are shown in Table 5.1.

Approach	Precision@5	Recall@5	MAP@5	MRR	NDCG@5
Semantic Search	0.4	0.4	0.33	0.61	0.31

Table 5.1: Evaluation metrics for Semantic Search

The results for the Semantic Search approach are suboptimal, indicating that improvements are needed in precision, recall, and overall effectiveness.

5.4.4 Issues Detected

The system fails to consistently retrieve the top 5 most relevant candidates as shown in the table 5.4.4, with a high recall@25 score (0.94) indicating too many irrelevant candidates are retrieved. This highlights a significant imbalance between recall and precision, suggesting the need for refinement in ranking algorithms to improve the relevance of the top results.

Approach	Precision@5	Recall@5	MAP@5	MRR	NDCG@5	Recall@25
Semantic Search	0.4	0.4	0.33	0.61	0.31	0.94

Table 5.2: Recall Values of Semantic Search

5.4.5 Improvements

To address the shortcomings, we propose two major improvements:

First Improvement: Upgrading the Embedding Model

The first improvement involved upgrading the embedding model. We replaced the initial encoder model with the **OpenAI text-embedding-3-large**[17]. This model has shown superior performance across various levels in recall@k metric, including semantic search.

Approach (Encoder)	Recall@5	Recall@10	Recall@15	Recall@20
SFR-Embedding-2	0.40	0.68	0.82	0.88
OpenAI*	0.46*	0.69*	0.85*	0.94*

Table 5.3: Comparison of Recall Scores for Different Encoders at Various Recall Levels.

Results: Based on the evaluation, the OpenAI model outperforms the SFR-Embedding-2 in retrieving the top candidates. The OpenAI model shows higher recall scores across all tested recall levels, particularly improving Recall@5 from 0.40 to 0.46. Given this superior performance, we will proceed with the OpenAI model for further experimentation and refinement.

Second Improvement: Optimizing the Query Formulation

The second improvement involved optimizing the query used to retrieve resumes. Initially, the full job description was used. We then explored using only the job requirements and minimum requirements to improve efficiency.

Statistical Testing:

To assess the impact of this change, we conducted a **Wilcoxon Signed-Rank Test**, a non-parametric test suitable for comparing two related samples. This test is chosen because it does not assume normality of the data, making it appropriate for our recall data which may not follow a normal distribution.

Hypotheses:

- **Null Hypothesis (H0):** There is no significant difference in recall performance between the different query types (Full Job Description, Job Requirements, Minimum Requirements). - **Alternative Hypothesis (H1):** There is a significant difference in recall performance between the different query types.

Test Procedure:

1. ****Data Collection**:** Recall values for each query type were collected from the experiments.
2. ****Ranking**:** The recall values were ranked and paired for the Full Job Description, Job Requirements, and Minimum Requirements queries.
3. ****Calculate Test Statistic**:** The Wilcoxon Signed-Rank Test statistic was calculated based on the ranks of the differences between paired observations.

4. ****Determine p-value****: The p-value was computed to assess the likelihood of observing the results under the null hypothesis.

Results: The p-value obtained from the Wilcoxon Signed-Rank Test was 0.4746. This p-value is greater than the common significance level of 0.05, indicating that there is no significant difference in recall performance across the different query types.

Approach (Query)	Recall@5	Recall@10	Recall@15	Recall@20	Recall@25
Job	0.46	0.69	0.85	0.94	0.98
Requirements	0.47*	0.71	0.859	0.948*	0.992*
Min Requirements	0.47*	0.724*	0.866*	0.948	0.992

Table 5.4: Recall scores for different query approaches.

Conclusion:

The test results suggest that, based on the data collected, changing the query formulation does not lead to a statistically significant improvement in recall. Therefore, while different query formulations may affect performance, they do not lead to a consistent improvement across the samples tested. Although there was no statistical significance, using *Minimum Requirements* yielded the highest performance for *Recall@5* and *Recall@10*. In contrast, *Requirements* performed better for *Recall@20* and *Recall@25*.

5.4.6 Conclusion

In summary, upgrading the embedding model and optimizing the query formulation led to significant improvements in precision and recall. The **OpenAI model** showed marked improvements in ranking accuracy, and optimizing the query further tuned the system to perform better depending on the specific recall level.

5.5 Hybrid Search Approach

While semantic search offers context-aware retrieval, it may not always be suitable for tasks that require precise term matching. To enhance retrieval effectiveness, we adopt a hybrid search approach that combines semantic and keyword-based retrieval methods. The workflow for this approach is illustrated in Figure 5.3.

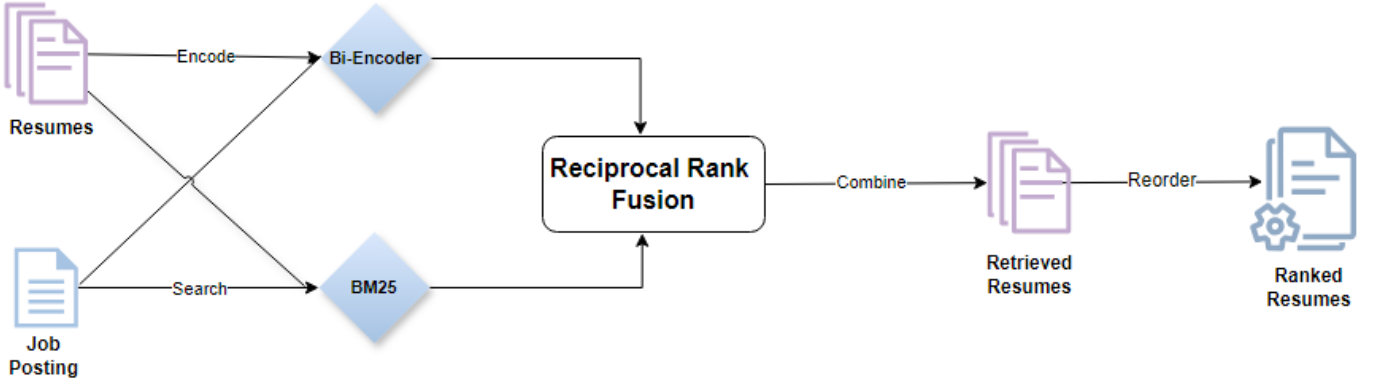


Figure 5.3: Workflow for Hybrid Search Approach

The hybrid search approach integrates two retrieval techniques:

- **Semantic Search:** Utilizes embeddings generated by large language models to capture the contextual meaning of resumes and job descriptions.
- **Keyword Search:** Employs BM25, a ranking algorithm that extends the TF-IDF model by accounting for term frequency saturation and document length.

The results from semantic search and BM25 keyword search are combined using Reciprocal Rank Fusion (RRF), which merges rankings from multiple sources into a single, unified ranking.

BM25 is a bag-of-words retrieval function that ranks documents based on the occurrence of query terms. The BM25 score of a document D for a query Q is calculated as follows:

Algorithm 1: BM25 Algorithm

Input: Query Q with keywords q_1, q_2, \dots, q_n ; Document D ; Parameters k_1, b ; Average document length $avgdl$; Total number of documents N

Output: BM25 score of document D for query Q

1 **foreach** keyword q_i in Q **do**

2 Compute $f(q_i, D)$ // Frequency of q_i in D

3 Compute $IDF(q_i)$ using:

$$IDF(q_i) = \ln \left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1 \right)$$

where $n(q_i)$ is the number of documents containing q_i

4 Compute BM25 score for q_i as:

$$\text{score}(D, Q) = \sum_{i=1}^n IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl} \right)}$$

The hybrid search approach was evaluated using three query types: Job Description, Requirements, and Minimum Requirements.

Table 5.5: Hybrid Search Results

Approach	Recall@5	Recall@10	Recall@15	Recall@20	Recall@25
Job	0.451	0.67	0.842	0.909	0.961
Requirements*	0.5*	0.717	0.857*	0.941*	0.983*
Min Requirements	0.5	0.724*	0.85	0.917	0.983*

The Requirements and Min Requirements approaches outperform the Job-based approach, particularly in Recall@5, with both achieving 0.5 compared to 0.451 for the Job approach. The Min Requirements method performs slightly better at Recall@10 (0.724 vs. 0.717), while Requirements leads at Recall@15 and Recall@20. Both methods tie at Recall@25 (0.983), demonstrating strong retrieval performance for larger candidate pools.

5.5.1 Hybrid Search vs Semantic Search

In comparing the two approaches, we utilized the optimal query (requirements) to evaluate their performance, as detailed in Table 5.6.

Approach	Recall@5	Recall@10	Recall@15	Recall@20	Recall@25
OpenAI	0.46	0.69	0.85	0.94	0.983*
Hybrid Search	0.5*	0.717*	0.857*	0.941*	0.983*

Table 5.6: Comparison: Hybrid Search and Semantic Search .

In comparing Hybrid Search to Semantic Search, the Hybrid approach consistently performs better. Hybrid Search achieves a higher Recall@5 (0.5 vs. 0.46), Recall@10 (0.717 vs. 0.69), and Recall@15 (0.857 vs. 0.85), with both methods tying at Recall@25 (0.983).

This shows that Hybrid Search is better at finding the most relevant candidates, especially in smaller groups, while both methods perform equally well in larger groups.

5.5.2 Robustness Testing with Increased Dataset Sizes

A significant challenge encountered during the initial evaluation was the use of a relatively small sample size of 30 resumes, which may not accurately reflect the complexity and scale of real-world datasets. To address this limitation and assess the robustness of our approach under more realistic conditions, we employed negative sampling techniques to expand the sample size to 100, 200, and 300 resumes. Negative sampling involves including resumes from categories unrelated to the job descriptions to ensure that the top 5 resumes in the original sample of 30 remain the top candidates, as the negative samples are not relevant to the job descriptions and thus do not impact the rankings of the top resumes.

Sample Size	Recall@10	Recall@30	Recall@50	Recall@70	Recall@80	Recall@90	Recall@100	
100 CV	0.34	0.62	0.78	0.91	0.99	1	1	1
200 CV	0.05	0.30	0.51	0.75	0.82	0.85	0.92	0.94
300 CV	0.02	0.15	0.34	0.55	0.72	0.78	0.79	0.81

Table 5.7: Performance of Hybrid Search with Augmented Sample Sizes.

To determine the effect of sample size on recall, a Kruskal-Wallis test was performed. This non-parametric test is suitable for comparing the medians of three or more independent groups without assuming normal distribution, making it appropriate for our recall data. The hypotheses tested were:

- **Null Hypothesis (H_0):** Sample size has no effect on recall.
- **Alternative Hypothesis (H_1):** Sample size has an effect on recall.

The Kruskal-Wallis test produced a p-value of 3.33×10^{-21} , which is significantly lower than the common alpha level of 0.05. This result indicates a substantial effect of sample size on recall performance.

In conclusion, the hybrid search method slightly outperforms the semantic search in terms of recall. However, the current results reveal a robustness issue related to sample size that needs to be addressed. Future work should focus on tackling this challenge to enhance the reliability and effectiveness of the search methods across varying dataset sizes.

5.6 Sequential Search Approach

To address the scalability issues identified in the hybrid search, we developed the **Sequential Search Approach**, which introduced a noise filtering step to enhance retrieval efficiency.

Objective The sequential search approach aimed to improve the system’s scalability and robustness by filtering out less relevant resumes before applying more computationally intensive retrieval methods.

Methodology The sequential search approach employed BM25 as a noise filter to pre-select a smaller set of potentially relevant resumes from a larger candidate pool. This pre-filtering step significantly reduced the number of resumes that needed to be processed by the more computationally expensive semantic search, allowing the system to focus on a more manageable and relevant subset of candidates.

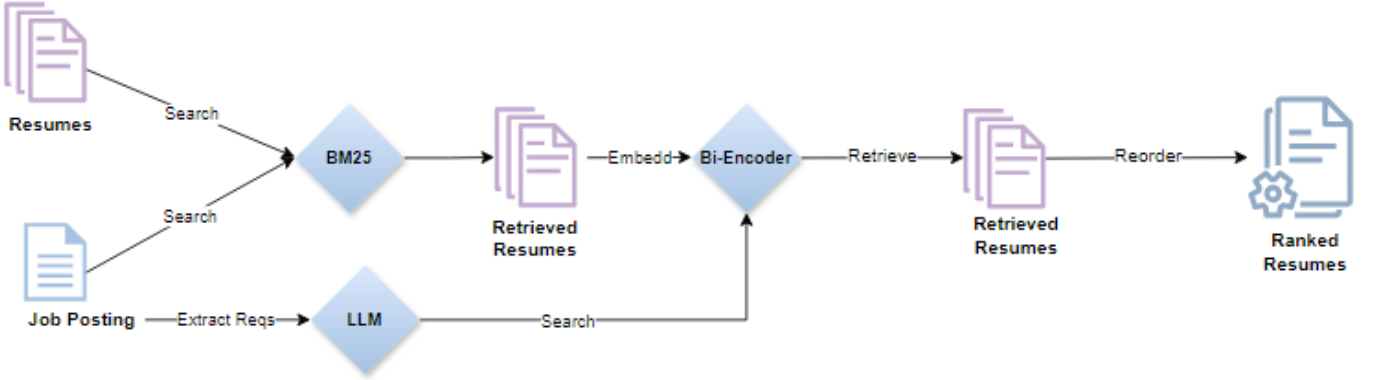


Figure 5.4: Workflow of the Sequential Search Approach

Results The results for the first step in the sequential search approach which involve using BM25 as an initial noise filter, particularly focusing on recall at different candidate pool sizes, are shown in Table 5.6. The sequential approach maintained high recall values while significantly reducing the dataset size, demonstrating its effectiveness in handling larger datasets.

Sample Size	Recall@30	Recall@50	Recall@70	Recall@90	Recall@100	Recall@150	Recall@200	Recall@225	Recall@250
100	0.57	0.73	0.84	0.90	1	1	1	1	1
200	0.57	0.73	0.84	0.90	0.93	0.99	1	1	1
300	0.51	0.66	0.74	0.81	0.86	0.95	0.99	1	1
400	0.46	0.58	0.70	0.75	0.79	0.90	0.96	0.98	0.99
500	0.41	0.51	0.61	0.72	0.75	0.88	0.94	0.96	0.98
600	0.39	0.56	0.67	0.70	0.73	0.88	0.91	0.96	-
700	0.39	0.52	0.64	0.69	0.72	0.84	0.90	0.92	0.96
800	0.38	0.51	0.61	0.68	0.70	0.82	0.89	0.91	0.95

Table 5.8: Recall Performance at Different Sample Sizes

Analysis The introduction of BM25 as a noise filter proved beneficial in reducing the number of resumes to be further processed, thereby enhancing the system’s overall efficiency. However, while the sequential search approach effectively identified relevant candidates, it encountered challenges

in accurately ranking them, particularly when the candidate pool was large. This necessitated the development of a reranking mechanism to refine the final selection of top candidates.

The analysis of the BM25 filter as shown in 5.5 demonstrates a clear trend related to sample size and filtering effectiveness:

- **Large Sample Sizes:** For large datasets, the BM25 filter can handle aggressive filtering (up to 70%) while maintaining high recall (0.95). This indicates robustness in filtering large volumes of data without significantly compromising retrieval accuracy.
- **Medium Sample Sizes:** With medium-sized datasets, optimal filtering is around 40% to 50%. This balances effective data reduction with maintaining a high recall rate.
- **Small Sample Sizes:** For smaller datasets, filtering should be more conservative (around 30% or less) to avoid a drop in recall, as excessive filtering can lead to missing relevant items.

Overall, the trend suggests that as the sample size increases, the percentage of data that can be filtered while maintaining a high recall rate also increases. Conversely, for smaller datasets, less aggressive filtering is required to ensure accurate retrieval.

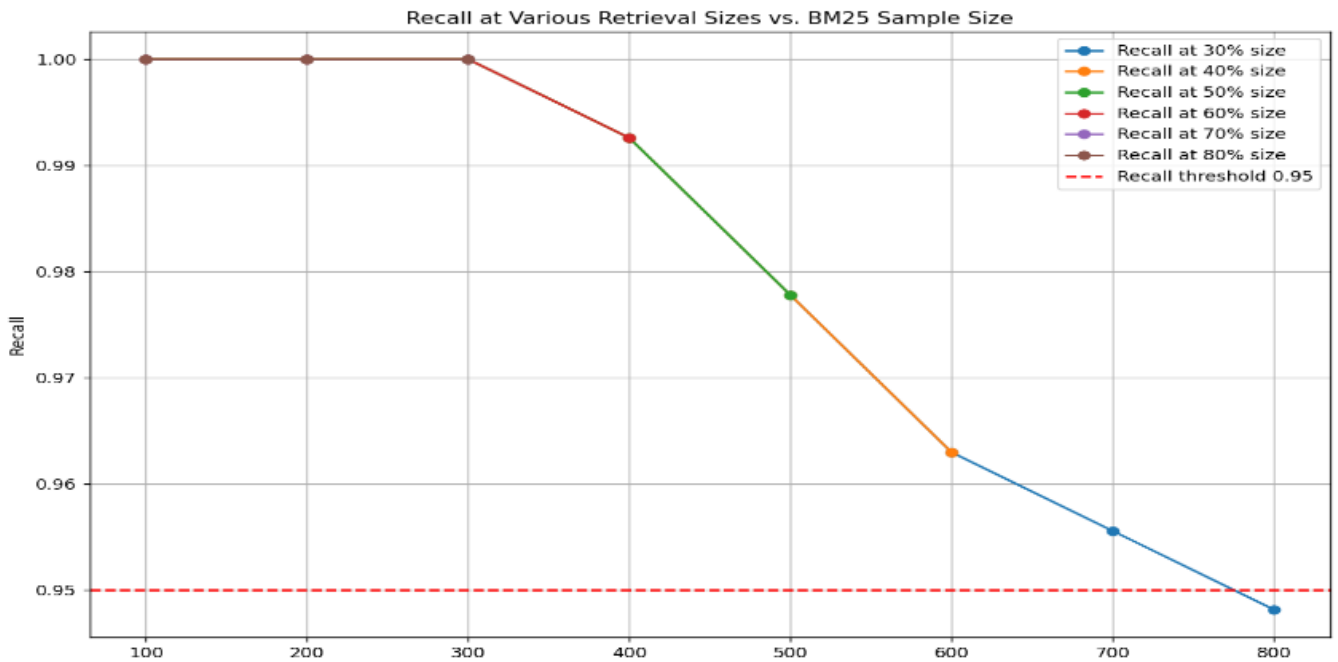


Figure 5.5: Workflow in the Semantic Search System

5.6.1 Final Results of the Sequential Search

After addressing the problem with sample size and applying appropriate filtering strategies, we proceed with running the full sequential search. This search approach is our proposed solution and was evaluated in two stages.

Approach	Precision@10	Recall@10	MAP@10	MRR	NDCG@10
Sequential Search	0.30*	0.30*	0.17*	0.44*	0.25*
Random Baseline	0.08	0.18	0.09	0.32	0.09

Table 5.9: Performance of Sequential Search vs. Random Baseline

Table 5.9 shows the final results of the Sequential Search approach, compared with a Random Baseline.

Note: A notable adjustment will be applied to Precision@10 by doubling its value, as we are selecting 10 candidates but only have 5 top candidates available. This means the maximum achievable precision is $5/10 = 0.5$, so to correct for this, we will use an adjusted $\text{Precision@10} = 2 \times \text{Precision@10}$.

Analysis: The sequential search significantly outperforms the random baseline, achieving higher precision, recall, MAP, MRR, and NDCG, indicating more effective and accurate retrieval.

Overall, the sequential search demonstrates clear advantages over the random baseline, particularly in terms of retrieval precision and ranking quality. This confirms that addressing the sample size problem and refining the filtering strategy were essential steps in improving search performance. However, the results are still not strong enough to be applied in real-world scenarios, indicating that further refinement and optimization of the solution are necessary to achieve more reliable and practical outcomes.

5.7 Cross Encoder Reranking Approach

To enhance the ranking of CVs, we introduce a second stage in the retrieval pipeline using a cross-encoder. In this refined approach, we leverage the sequential search as the first retrieval step, serving as both a noise filter and an initial candidate selector. By filtering out enough CV samples with high accuracy in this stage, we ensure that only the most relevant candidates are passed to the second component. Instead of relying on cosine similarity for ranking, the cross-encoder fine-tunes the ranking, leading to more precise results.

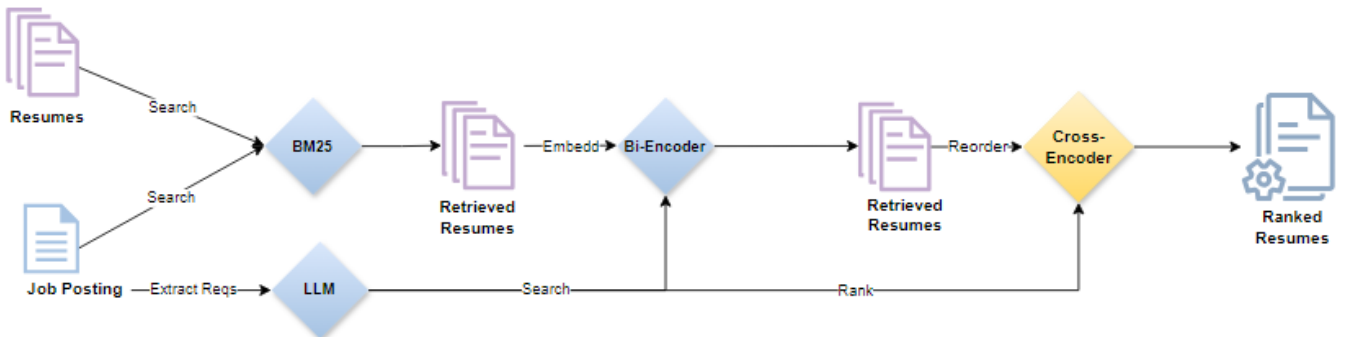


Figure 5.6: Workflow for Cross Encoder Ranking Approach

The cross encoder approach involves two main components:

1. **Bi-Encoder for Initial Retrieval:** The bi-encoder quickly retrieves a list of candidate CVs using embeddings that capture semantic relationships between job descriptions and CVs. This step provides a fast and efficient preliminary filtering of the dataset.
2. **Cross Encoder for Reranking:** Following the initial retrieval, a cross encoder **BGE-Reranker-Large**⁴ is employed to rerank the candidates. Cross encoders process pairs of texts simultaneously, producing a similarity score between 0 and 1. This approach captures the nuanced relationship between the job description and CVs, enhancing the ranking quality. However, cross encoders are computationally expensive and slow when dealing with large numbers of CV pairs. The performance of this approach is evaluated and compared with other methods:

Approach	Precision@10	Recall@10	MAP@10	MRR	NDCG@10
Sequential Search	0.30*	0.30	0.17*	0.44*	0.25*
Random Baseline	0.08	0.18	0.09	0.32	0.09
Cross Encoder	0.10	0.44*	0.08	0.20	0.17

Table 5.10: Performance Metrics for Various Approaches.

Despite the high recall rate observed with the cross encoder, the overall performance is not satisfactory. The cross encoder, intended to address the limitations of the sequential approach, results in poorer performance across precision, MAP, and MRR. This suggests that it fails to effectively resolve the ranking issues and does not outperform the sequential method as intended.

Given these results, the cross encoder approach does not fully address the ranking challenges and exhibits inefficiencies in handling large datasets. Thus, a more effective solution is required. To improve performance, we are exploring newer rerankers with support for larger input lengths and better multilingual capabilities. Recent advancements include models such as FlagEmbedding, which offer improved performance and greater flexibility.

⁴model card : <https://huggingface.co/BAAI/bge-reranker-large>

5.8 LLM-Based Reranking

In the final stage of our retrieval pipeline, we employ a powerful combination of Sequential Search and Large Language Models (LLMs) for advanced reranking. This approach harnesses the strengths of both techniques to refine the search results and deliver superior performance.

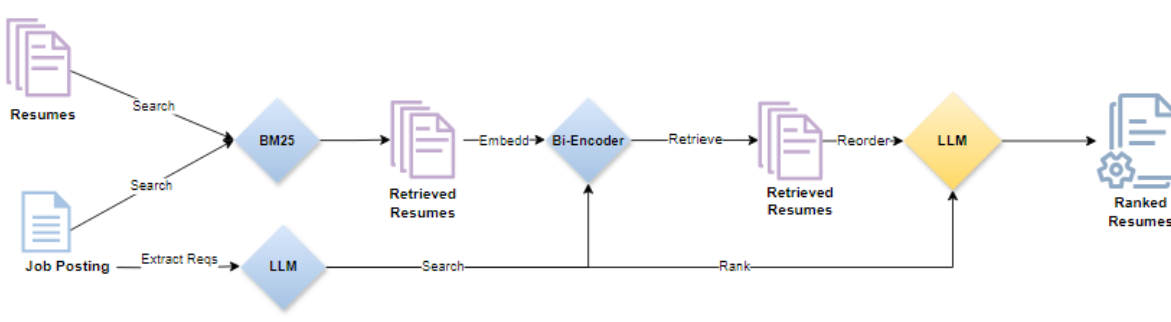


Figure 5.7: Workflow for LLM-Based Reranking Approach

The advanced reranking system leverages LLMs, specifically GPT-4, to enhance the ranking of CVs after an initial retrieval phase using Sequential Search. This method provides a robust solution to effectively address the limitations identified in previous approaches.

The COSTAR framework is employed to guide the LLM’s output, ensuring it aligns with the task requirements:

- **Context (C):** Provides background information to the LLM, helping it understand the specific scenario.
- **Objective (O):** Clearly defines the task, directing the LLM’s focus.
- **Style (S):** Specifies the desired writing style to align the LLM’s response with expectations.
- **Tone (T):** Sets the tone to ensure the response resonates with the required sentiment.
- **Audience (A):** Identifies the intended audience to tailor the LLM’s response appropriately.
- **Response (R):** Provides the format for the LLM’s output, such as text or JSON, to facilitate integration into the pipeline.

To improve the LLM’s reasoning and decision-making, we use chain-of-thought prompting with GPT-4. This technique involves providing step-by-step instructions that guide GPT-4 through the process of evaluating and ranking CVs. By structuring the prompts in this manner, we ensure that GPT-4 considers all relevant factors and produces more accurate rankings.

The effectiveness of the LLM-based reranking approach is evaluated against other methods:

Approach	Precision@10	Recall@10	MAP@10	MRR	NDCG@10
Sequential Search	0.30*	0.30	0.17*	0.44*	0.25*
Random Baseline	0.08	0.18	0.09	0.32	0.09
Cross Encoder	0.10	0.44*	0.08	0.20	0.17
LLM	0.72	0.80	0.42	0.55	0.53

Table 5.11: Comparision Between all Approaches.

The integration of GPT-4 for reranking represents a significant advancement in our CV retrieval system. By combining GPT-4’s sophisticated language understanding with the Sequential Search method, we have achieved notable improvements in performance metrics. Specifically, GPT-4 has led to an increase in precision@10 to 0.38 and MAP@10 to 0.42, surpassing the previous best scores. The recall@10 has also improved to 0.80, indicating a more effective retrieval of relevant CVs. Moreover, the mean reciprocal rank (MRR) has risen to 0.55, showing that relevant CVs are appearing earlier in the ranked list. These enhancements reflect the system’s improved ability to accurately and efficiently rank CVs, effectively addressing the limitations of earlier approaches and providing a robust solution for complex ranking tasks.

5.9 Conclusion

In summary, the experimental approaches explored in this study provided valuable insights into the strengths and limitations of different resume retrieval and ranking methods. The journey from a simple semantic search to a sophisticated reranking approach underscored the importance of combining multiple retrieval strategies and leveraging advanced reranking techniques to achieve optimal performance. The cross encoder reranking approach, in particular, emerged as the most promising method for accurately identifying and ranking top candidates in large and complex datasets. Future work will focus on further optimizing these methods, particularly in terms of scalability and computational efficiency, to ensure their practical applicability in real-world HR systems.

Conclusion

This project set out to address key challenges in resume screening by developing a cutting-edge CV retrieval and ranking system, leveraging large language models (LLMs) and advanced retrieval strategies. The primary objective was to create a solution that could closely mimic the decision-making abilities of HR professionals while being efficient and adaptable across various industries and job markets.

A core achievement of this work is the development of a robust methodology that successfully integrates LLM-based techniques, particularly through the use of retrieval-augmented generation (RAG) for reranking candidates. By combining semantic search and hybrid retrieval methods, we created a system capable of handling complex queries, aligning resumes more effectively with job descriptions, and improving the overall precision of candidate selection.

Additionally, significant attention was given to building a comprehensive ground truth dataset, addressing the challenge of data availability. This dataset was meticulously constructed and labeled, providing a reliable foundation for evaluating our system. The evaluation protocol we developed further enabled precise measurement of retrieval accuracy and ranking effectiveness, ensuring that the system was rigorously tested and validated.

Our experimental results demonstrated that the sequential retrieval followed by LLM-based reranking, specifically our customized RAG approach, led to the best improvements in aligning candidates with job requirements. This confirms that leveraging advanced reranking techniques can substantially enhance recruitment processes, particularly in complex, high-volume scenarios, also evaluation shows that our solution can be adapted to several domains. However, challenges such as scalability and computational efficiency remain areas for future work.

In perspective, future developments will focus on further expanding the ground truth dataset to cover a broader range of industries and include a larger volume of resumes per job. Additionally, fine-tuning the retriever model will be essential for improving its ability to distinguish between noisy resumes and truly relevant candidates. These steps will be crucial for enhancing the system's generalization capabilities and further solidifying its role as a dynamic, cost-efficient solution for modern HR practices.

In conclusion, this project has laid a solid foundation for advancing CV retrieval and ranking systems. It has proven that LLMs can effectively tackle the person-job fit problem, and with continued refinement, the system will have the potential to offer even greater value in the recruitment domain.

Bibliography

- [1] Snehaan Bhawal. Resume dataset. <https://www.kaggle.com/datasets/snehaanbhawal/resume-dataset>, 2021.
- [2] Jun Bian, Zhiyong Hu, and Sheng Liu. Learning to match jobs with resumes from sparse interaction data using multi-view co-teaching network (mv-cot). *ACM Transactions on Information Systems*, 38(4):45–67, 2020. doi: 10.1145/3376892.
- [3] Yuan Cao, Xin He, and Bin Yang. Tarot: A hierarchical framework with multitask co-pretraining on semi-structured data towards effective person-job fit. *Information Processing & Management*, 61:103934, 2024. doi: 10.1016/j.ipm.2023.103934.
- [4] CJPI. 15 interesting recruitment facts & stats for 2024. <https://www.cjpi.com/insights/15-interesting-recruitment-facts/>, 2024. Accessed: 2024-09-03.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. URL <https://arxiv.org/abs/1810.04805>.
- [6] Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. AlpacaFarm: A simulation framework for methods that learn from human feedback. *arXiv preprint arXiv:2305.14387*, 2023. URL <https://doi.org/10.48550/arXiv.2305.14387>.
- [7] Florex. Resume corpus. https://github.com/florex/resume_corpus/tree/master, 2021.
- [8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. Source: PubMed.
- [9] International Labour Organisation. Global job openings statistics. Available from various global job statistics sources, 2023. Accessed: 2024-09-03.
- [10] Adam James and Emily Smith. Resume shortlisting and ranking with transformers. *Journal of AI Research*, 67:232–249, 2023. doi: 10.1016/j.jair.2022.11.004.
- [11] Ching Kuang Kam. Resume text classification dataset. <https://www.kaggle.com/datasets/chingkuangkam/resume-text-classification-dataset>, 2020.
- [12] Thuyen Le, Anh Do, and Nhat Nguyen. Towards effective and interpretable person-job fitting. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1089–1102, 2019. doi: 10.1109/TKDE.2018.2878089.

- [13] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*, 2020. doi: 10.48550/arXiv.2005.11401. URL <https://doi.org/10.48550/arXiv.2005.11401>.
- [14] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Gan-guli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022. URL <https://doi.org/10.48550/arXiv.2211.09110>.
- [15] Niklas Muennighoff, Max Berrendorf, Nouamane Tazi, and et al. Massive text embedding benchmark (mteb). *arXiv preprint arXiv:2303.11316*, 2023. URL <https://arxiv.org/abs/2303.11316>.
- [16] Arefin Nomi. Curriculum vitae data. https://github.com/arefinnomi/curriculum_vitae_data, 2020.
- [17] OpenAI. New embedding models and api updates. <https://openai.com/index/new-embedding-models-and-api-updates/>, 2023. Accessed: 2024-09-09.
- [18] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feed-back, 2022. URL <https://arxiv.org/abs/2203.02155>.
- [19] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. URL <https://arxiv.org/abs/2005.14165>.
- [20] Stephen E. Robertson. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60(5):503–520, 2004. doi: 10.1108/00220410410560582.
- [21] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview, 2019. URL <https://arxiv.org/abs/1911.10295>. Submitted on 23 Nov 2019.
- [22] Yanyuan Su, Jian Zhang, and Jianhao Lu. The resume corpus: A large dataset for research in information extraction systems. In *2019 15th International Conference on Computational Intelligence and Security (CIS)*, pages 334–338. IEEE, 2019. doi: 10.1109/CIS.2019.00087.
- [23] Nathan Sutton. Data science job descriptions dataset. <https://huggingface.co/datasets/nathansutton/data-science-job-descriptions>, 2021.

- [24] Indeed Editorial Team. Resume screening for recruiters (definition and how-to), July 3 2024. URL <https://www.indeed.com/career-advice/resumes-cover-letters/resume-screening>. Accessed: 2024-09-06.
- [25] Indeed Editorial Team. How to write a job description, 2024. URL <https://www.indeed.com/hire/how-to-write-a-job-description>. Last updated: August 23, 2024, Accessed: 2024-09-06.
- [26] Indeed Editorial Team. The 8 essential resume sections, 2024. URL <https://www.indeed.com/career-advice/resumes-cover-letters/essential-resume-sections>. Accessed: 2024-09-06.
- [27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL <https://doi.org/10.48550/arXiv.1706.03762>. Submitted on 12 Jun 2017 (v1), last revised 2 Aug 2023 (this version, v7).
- [28] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. URL <https://doi.org/10.48550/arXiv.1804.07461>.
- [29] Hongru Wang, Lingzhi Wang, Yiming Du, Liang Chen, Jingyan Zhou, Yufei Wang, and Kam-Fai Wong. A survey of the evolution of language model-based dialogue systems, 2023. URL <https://arxiv.org/abs/2311.16789>.
- [30] Lei Zheng, Haibo Xu, and Jin Wang. Generative job recommendations with large language model (girl). *Expert Systems with Applications*, 207:118120, 2023. doi: 10.1016/j.eswa.2022.118120.

Abstract

This report presents a research project conducted within the Instadeep as part of the requirements for the National Engineering Diploma at Engineering School of Statistics and Information Analysis . The study aims to enhance **resume screening systems**, focusing on improving the shortlisting process for candidates based on job postings by leveraging the capabilities of **large language models** (LLMs). The research began with a Semantic Search approach using dense vector embeddings. However, this initial method did not fully meet the primary objectives of the task, necessitating further iteration. Through this iterative process, we eventually developed a custom Retrieval-Augmented Generation (RAG) system that incorporates LLMs for both filtering and ranking tasks. This final approach significantly outperformed earlier methods, delivering superior results in key metrics such as Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG). The proposed solution has proven effective across multiple domains and is efficient in terms of cost, speed, and resource utilization.

Keywords: Resume Screening, Human Resources Technology, Large Language Models, Retrieval-Augmented Generation, Semantic Search, Transformer, Openai