Candidate Profile Summarization- A RAG Approach with Synthetic Data Generation for Tech Jobs

Anum Afzal
Technical University of Munich
anum.afzal@tum.de

Ishwor Subedi
Technical University of Munich
ishwor.subedi@tum.de

Florian Matthes
Technical University of Munich
matthes@tum.de

Abstract

As Large Language Models (LLMs) become increasingly applied to resume evaluation and candidate selection, this study investigates the effectiveness of using in-context example resumes to generate synthetic data. We compare a Retrieval-Augmented Generation (RAG) system to a Named Entity Recognition (NER)-based baseline for job-resume matching, generating diverse synthetic resumes with models like Mixtral-8x22B-Instruct-v0.1. Our results show that combining BERT, ROUGE, and Jaccard similarity metrics effectively assesses synthetic resume quality, ensuring the least lexical overlap along with high similarity and diversity. Our experiments show that RAG notably outperforms NER for retrieval tasks—though generation-based summarization remains challenged by role differentiation. Human evaluation further highlights issues of factual accuracy and completeness, emphasizing the importance of in-context examples, prompt engineering, and improvements in summary generation for robust, automated candidate selection.

1 Introduction

The increasing demand for intelligent recruitment solutions has driven the development of automated systems for resume parsing, candidate-job matching, and talent recommendation. However, a persistent challenge in building such systems is the limited availability of high-quality, annotated datasets, primarily due to the sensitive and private nature of resume and recruitment data. Additionally, the diversity in resume formats, terminologies, and job description structures further complicates the training of generalizable machine learning models.

To address this data scarcity, synthetic data generation has emerged as a promising alternative (Nadăș et al., 2025; Long et al., 2024; Li et al., 2023; Bauer et al., 2024). By creating realistic, diverse, and well-structured synthetic resumes and job descriptions, researchers can augment existing datasets or build entirely new corpora for training robust natural language processing (NLP) models. Recent advances in large language models (LLMs) and generative frameworks have greatly enhanced the realism and semantic coherence of such synthetic documents, making them viable for downstream applications such as candidate screening, resume classification, and job matching.

Simultaneously, Retrieval-Augmented Generation (RAG) has gained traction as an effective architecture for tasks requiring grounded text generation and semantic understanding. In the recruitment domain, RAG is well-suited for modeling the resume-job description matching problem, allowing relevant information to be retrieved and synthesized from both structured and unstructured textual sources (Gan et al., 2024). Unlike traditional classification-based approaches, RAG enables dynamic and explainable matching by leveraging both information retrieval from candidate or job corpora and generative reasoning. While many companies have developed efficient in-house Applicant Tracking Software (ATS) to attract top talent, open-source datasets that support general research in this domain remain scarce.

In this paper, we explore the combined use of RAG and synthetic data generation with LLMs to improve recruitment systems (Lewis et al., 2020b). Specifically, we focus on generating synthetic resume data and applying RAG for candidate retrieval and summarization in response to job descriptions. Our objectives are to determine whether synthetic data can be generated to match the distribution of real-world resumes (Chim et al., 2025) and to evaluate the effectiveness of RAG-based retrieval compared to traditional Named Entity Recognition (NER) approaches for job-resume matching. Our key contributions are as follows:

- Propose a method for generating synthetic resume data that closely aligns with the distribution and characteristics of real-world resumes.
- Develop a framework for evaluating the quality and realism of the generated synthetic resumes.
- Investigate the effectiveness of the RAG-based approach for candidate selection, benchmarking its performance against a traditional NER baseline.
- Evaluate the capability of LLMs to summarize resumes in the context of specific job descriptions.

2 Related Work:

2.1 Synthetic Data Generation:

In the context of synthetic data generation, some older approaches generate variation in words, and synonyms were used from the WordNet (Miller, 1995). More recent approaches involving LLMs, including Skondras et al. (2023), generate synthetic resumes with ChatGPT for the task of job description classification. In order to generate variation in the job, they use different writing styles (e.g, Scientific, literary, colloquial). These variations were generated with prompt engineering. Furthermore, similar to our work Li et al. (2023) also utilize LLMs for synthetic data generation, but for a text classification task. Chim et al. (2025) design an LLM-driven framework for both synthetic data generation and evaluation. Younes et al. (2025) present an LLM-driven end-to-end pipeline for all tasks of an ATS. Lastly, Lo et al. (2025) presents a similar but explainable ATS system.

2.2 Resume-Job Matching

Application Tracking systems have been the backbone for companies to ease the recruiting process for years. However, their presence in academic literature is lacking, although not completely nonexistent. Otani et al. (2025) presents an extensive survey of how LLMs are used by Human Resource. Gan et al. (2024) demonstrated an approach where they use LLM agents to candidate application tracking and management. In terms of Resume to Job matching, traditional approaches employ a Name Entity Recognition (NER) and regex to extract skills and job titles from regex(Sougandh et al., 2023), which are matched to find the top candidate. One such approach is demonstrated by resumes(Warusawithana et al., 2023), who develop a layout-aware parsing of the resume based on the section of the resume. In combination with section-aware parsing, they use rule-based techniques to extract the details of the resume. Modern machine learning methods combine two techniques of parsing Resumes and combining it to develop a job recommender (Chandak et al., 2024), where they use Tf-idf vectorizer to find the similarity with cosine similarity metrics to find similar job recommendations.

2.3 Resume Summarization

Resume summarization is a very common task for HR monitoring and the recruitment process. There have also been several research studies on the application of LLMs (Gan et al., 2024). In (Upadhye, 2024), authors study the efficiency of various models such as BART (Lewis et al., 2020a), T5 (Raffel et al., 2023), and BERT (Devlin et al., 2018) in extracting various skills from the resume and summarizing it. Similarly, in (Mercan et al., 2023), the resumes summarization capability of several encoder-decoder language models is evaluated. The result in (Mercan et al., 2023) suggests that fine-tuning BART-Large outperforms other models. In our study, we also take into the context of the job description while generating the summaries, which haven't been explored before.

3 Methodology

When working with Applicant Tracking Systems (ATS), one major challenge researchers face is the lack of high-quality datasets containing pairs of resumes and matched job descriptions. In our methodology, we focus on the field of computer science and software engineering. In the first part of this paper, we explore the potential of Large Language Models (LLMs) to generate a synthetic dataset using existing "anchor" resumes through in-context learning. Our experiments investigate whether it is possible to generate new resumes that are contextually similar to the anchor resumes but exhibit minimal lexical overlap. Specifically, we ask: *Can synthetic resumes accurately reflect the characteristics of anchor resumes while introducing controlled variations within the same domain?*. This paper aims to assess the feasibility of constructing diverse and representative synthetic datasets for future ATS research by leveraging the pre-trained knowledge of LLMs. In the second part of our methodology, we assess the effectiveness of a Retrieval-Augmented Generation (RAG) system for candidate profile selection and summarization, in comparison to a basic Named Entity Recognition (NER)-based approach. In the RAG framework, the job description is used as a query for the retriever component to identify the top-k most suitable candidates. Subsequently, a generator (LLM) produces a summary of each candidate's profile, explaining why they may be a good fit for the job. We used OpenAI [1] and TogetherAI[2] for running inferences on LLMs. We explain the core concepts of our methodology below:

3.1 Data Collection

We collected anchor CVs from IT professionals to generate the synthetic dataset. To maintain a focused scope, resumes were gathered from individuals working as software engineers, full-stack developers, AI scientists, and machine learning engineers. Participants provided informed consent, acknowledging both the intended use of their resumes and the anonymization process specific to this project. Additionally, each participant selected a job description from a provided set to indicate the role best suited to their profile; this information is used later for evaluating RAG system performance. In total, 10 participants contributed 21 resumes, covering roles such as Senior Python Developer, Data Scientist, Machine Learning Engineer, AI Scientist, Full-Stack Developer, Node.js Developer, DevOps, Kubernetes Developer, and others. The collected resumes were anonymized and converted to JSON format. We identified education, work experience, personal projects, and skills as the most important information in each CV. The conversion from anonymized text to structured JSON format was accomplished using the LLM Mixtral-8x22B (AI, 2024b). These collected resumes served as in-context learning examples for generating synthetic resumes in our study.

3.2 Synthetic Data Generation and Evaluation

For the purpose of synthetic data generation, we conducted prompt engineering with the goal of producing synthetic CVs closely matching the anchor CVs in terms of education, skills, and experience,

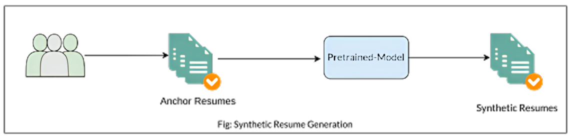

Figure 1: Pipeline for Synthetic Data Generation

while introducing enough variation to ensure they were not identical. Both the anchor CV and the target position title were included in the prompt to guide the LLM in generating the desired output. Manual prompt engineering was performed through an iterative process: prompts were crafted, resumes generated, and results evaluated, with adjustments made to improve alignment with real-world CV structure and content. The prompting focused on instructing the LLM to create resumes that were similar to the anchor examples with respect to domain, education, skills, and experience, while achieving the closest possible structural similarity to authentic CVs. This process required multiple iterations of manual prompt refinement. The four prompts demonstrated in Table 1 show the most significant results for further analysis. The evaluation of the synthetic resumes was conducted using two primary criteria: I) **Real world similarity** II) **Variation in synthetic resumes**.

- **Real-world similarity**: One key objective of synthetic data generation is to produce synthetic resumes that closely resemble real-world resumes, represented in our study by the anchor CVs. To assess this, we compare the synthetic resumes to the anchor CVs using ROUGE (Lin, 2004) and BERTScore (Zhang* et al., 2020). High semantic similarity in education, skills, and experience should result in high BERTScore; however, we also monitor for low ROUGE scores in n-gram overlap, indicating that while content is similar, exact phrasing and wording differ appropriately.

- **Variation in synthetic resumes**: In addition to similarity, it is important for each anchor CV to yield multiple, distinct synthetic resumes. To confirm sufficient diversity among synthetic resumes generated from the same anchor, we compute the 1-Jaccard similarity between these documents. The Jaccard similarity (Jaccard, 1901) measures the proportion of overlapping content, with lower values indi-

[1]https://openai.com/
[2]https://www.together.ai/

cating greater variation among the generated resumes.

3.3 Retrieval Augmented Generation

To assess the role of Retrieval-Augmented Generation (RAG) in candidate evaluation and selection, we implemented a RAG pipeline comprising a vector database for resume storage and a large language model (LLM) for summary generation (see Figure 2). Resumes in JSON format were encoded into embeddings using SentenceTransformer with the `Sentence-T5-base` model, chosen for its memory efficiency. These embeddings were stored in the FAISS[3] vector database (IndexFlatL2), offering lightweight, fast similarity search without the overhead of more complex vector databases. For retrieval, job descriptions were embedded in the same manner, and the top-K most similar resumes were selected based on L2 distance. The selected resumes were then summarized by Mixtral-8x22B-Instruct-v0.1, with prompt shown in Table 2 tailored to highlight key aspects such as skills, experience, and education. The relative importance (weights) of these resume sections was determined using LLM analysis of job descriptions, validated through human evaluation to ensure relevance and justification.

3.4 NER baseline:

The baseline approach uses Named Entity Recognition (NER) to extract skills from both job descriptions and resumes, ranking candidates according to their similarity with the job requirements. Resume text is first preprocessed using NLTK (Bird et al., 2009) for lemmatization and tokenization, and skills are then extracted with spaCy's (Honnibal and Montani, 2017) NER model (*en_core_web_sm*), enhanced by custom rules for accurate identification of technical skills such as Python, Java, and C++. After skill extraction, both resumes and job descriptions are vectorized with CountVectorizer, representing skill terms as token counts. Cosine similarity is then computed between these vectors to assess the alignment between each resume and the job description.

3.5 Evaluation

Retriever Evaluation The retriever component of the RAG pipeline is evaluated using several key metrics. Precision@k (Manning et al., 2008) measures the proportion of relevant resumes among the top-k retrieved documents by checking if the resumes share the same label as the job description, based on participant-provided ground truth. Recall@k (Manning et al., 2008) quantifies how many relevant resumes are retrieved within the top-k, indicating the system's ability to capture all suitable candidates. Mean Reciprocal Rank (MRR) evaluates the rank position of the first correct resume retrieved, while Accuracy assesses whether the very top-ranked resume is correct.

Generator Evaluation: The generator is evaluated in two ways: first, by measuring the accuracy of its selection of the best resume for a job description against ground truth labels; and second, through domain expert assessments of the generated summaries. Experts rate each summary on a 1–5 scale across four dimensions: relevance-how well key skills and experience are highlighted, completeness-whether all important information such as skills, experience, and education is included, clarity and conciseness-ease of understanding and structure, and accuracy-factual correctness of the summary compared to the original resume.

3.6 Experimental Settings

Synthetic Resume Generation: Initially, we experimented with synthetic data generation using GPT-4o-mini, Llama-3.1-70B-Instruct-Turbo (AI, 2024a), and Mixtral-8x22B-Instruct-v0.1. After assessing the performance of these models in early experiments, we selected Mixtral-8x22B-Instruct-v0.1 as our primary model due to its consistently superior output quality. For the generation, Mixtral was configured with a maximum token limit of 16,384, and the temperature setting was randomized to promote output diversity. For the baseline in synthetic data generation, we used **Prompt 1** and **Prompt 2** as standard prompts. Our evaluation compares the performance of these baseline prompts to prompts that include an in-context anchor resume, assessing how the inclusion of contextual examples influences the quality of generated synthetic data.

Retreival Augmented Generation: For the retriever component of the RAG system, resumes were semantically encoded using the SentenceTransformer (sentence-t5-base) (Reimers and Gurevych, 2019)model and stored in a FAISS (Facebook AI Similarity Search) (Douze et al., 2025) index for efficient similarity-based retrieval.

[3]https://faiss.ai/index.html

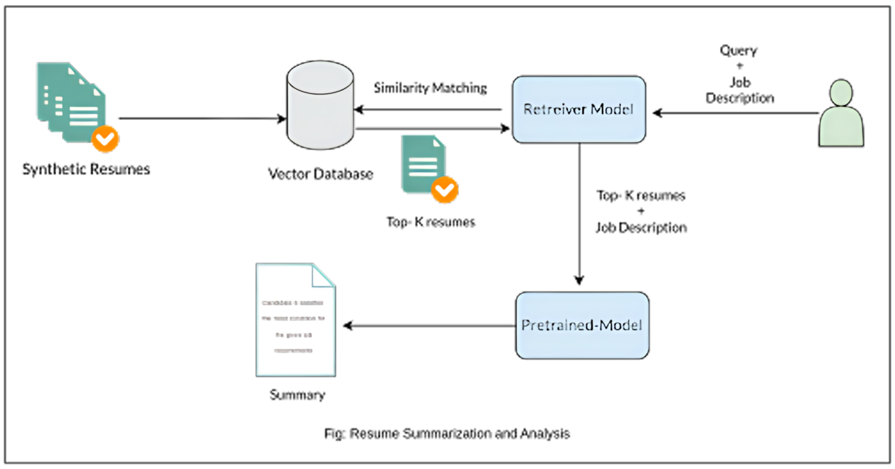

Figure 2: Retrieval Augmented Generation pipeline

Job descriptions were encoded in the same way, allowing the system to retrieve the top-k most similar resumes through vector similarity search. The generator component utilized Mixtral-8x22B-Instruct-v0.1 with the same settings as in the data generation phase, employing a summarization prompt provided in the appendices. As a baseline, the NER approach extracted named entities from resumes and job descriptions using a pretrained SpaCy[4] NER model, augmented with custom entity rules to recognize technical skills such as “JavaScript” and “Python” as SKILL entities. Following entity extraction, both resumes and job descriptions were vectorized using a TF-IDF vectorizer, and cosine similarity was computed to identify the top-k matching resumes for each job description.

4 Results and Discussions

4.1 Synthetic Resume Analysis:

For resume generation, several different prompts were tested, with the four most notable prompts detailed above. We initially utilized one-shot prompting with OpenAI’s GPT-4o-mini (OpenAI, 2024), which delivered adequate results but tended to overemphasize the job title specified in the prompt, rather than leveraging contextual examples. Next, we experimented with open-source models such as Llama-3.1-70B-Instruct-Turbo (AI, 2024a), but observed limited success: the outputs often overfit to the job title and struggled to follow explicit instructions. Additionally, the Llama model faced challenges related to context window size, which hindered its ability to incorporate one-shot and complete examples. Among all models evaluated, Mixtral-8x22B-Instruct-v0.1 (AI, 2024b) consistently produced the best results, and therefore all subsequent analyses are based on Mixtral outputs.

Comparing the two baseline prompts, Prompt 1 and Prompt 2, we found similar metric values, with the primary distinction being output format: Prompt 1 generated standard text, while Prompt 2 produced JSON-structured results as specified. Performance was evaluated by calculating BERTScore and ROUGE between anchor and synthetic resumes, as well as computing 1-Jaccard similarity to quantify diversity among synthetic resumes generated from the same prompt and anchor. For Prompt 2, even though the prompt excludes an explicit anchor resume, we used the resume associated with the applied job title as an anchor for evaluation purposes. The figures presented report the average metric values across all scenarios.

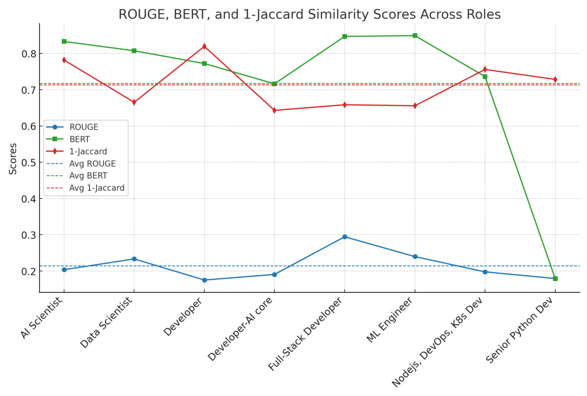

Figure 3: Results for the Synthetic Data Generation with Prompt 3

[4]https://spacy.io/

Generation Prompt 1

Generate a resume for {job_title}.

Generation Prompt 2

Generate a resume for the job_title. The generated resume should be in JSON format {example_json}. The output should just be a synthetic CV, nothing else.

Generation Prompt 3

You are an expert resume writer tasked with generating a synthetic resume. The generated resume must closely follow the structure, style, and patterns found in the example CV provided below. Use the following guidelines:

1. **Output Format:** The resume should be produced in the following JSON format: {example_json}
2. **Similarity to Example CV:**
    - The generated resume should mirror the organization, flow, and structure of the example CV, including section order (e.g., Work Experience, Projects, Education, Skills), the style and level of detail in descriptions, the types of roles, responsibilities, and projects are listed.
    - Work Experience: Include job titles, responsibilities, and achievements similar to those in the example CV, but modify company names, locations, and exact timeframes.
    - Maintain similar industries and roles but rephrase the descriptions to introduce uniqueness.
    - Projects: Reflect a similar scope, complexity, and style to the projects in the example CV. Include comparable tools, frameworks, or technologies, while changing project names and specific details.
    - Education: Keep the degree types and fields of study aligned with the example CV but vary institution names and years.
    - Skills and Certifications: List skills that align with the example CV, with slight additions or removals to create variation.
3. **Introduce Variations:**
    - While closely following the example CV's structure, introduce variations where appropriate. Replace names of companies, projects, and institutions with realistic alternatives.
    - Slightly modify timeframes and career progression while keeping them plausible.
    - Adjust tools, technologies, or techniques mentioned to provide diversity but ensure relevance to the example pattern.
    - Add or rephrase achievements and responsibilities to maintain uniqueness.
4. **Tone and Style:**
    - Maintain a professional tone that is consistent with the example CV.
    - Use concise bullet points for readability.

    Output the resume strictly in the JSON format specified above. The output should just be synthetic CV, nothing else. Here is the example CV that the generated resume should emulate in pattern and structure: {example_cv}

Generation Prompt 4

You are an expert resume writer tasked with generating a synthetic resume. The generated resume must closely follow the structure, style, and patterns found in the example CV provided below. Use the following guidelines while using the following CV as an example: {example_cv}

- **Similarity to Example CV:** Mirror the flow and structure of the example CV, including Section order (e.g., Work Experience, Projects, Education, Skills). The style and level of detail in descriptions. The types of roles, responsibilities, and projects are listed.
- **Work Experience:** Retain the same domain and industry focus but introduce diversity in roles, company types, and career trajectories (e.g., startups, multinational corporations, or government roles). Enrich job descriptions with unique responsibilities and achievements relevant to the domain but distinct from the example CV. Incorporate diverse metrics, tools, or frameworks to highlight different career impacts (e.g., specific revenue increases, customer satisfaction improvements, or technical implementations).
- **Projects:** Reflect a similar scope, complexity, and style to the projects in the example CV while diversifying themes (e.g., emphasize different goals, technologies, or outcomes). Change the focus of individual projects slightly to reflect broader expertise in the domain (e.g., if the example involves ”e-commerce optimization,” introduce ”healthcare tech” or ”supply chain systems” as variations).
- **Education:** Use different but plausible institutions, fields of study, and years. Add unique academic achievements or extracurricular involvements for variation (e.g., leadership roles, research papers, or exchange programs).
- **Skills and Certifications:** Introduce domain-relevant but distinct skills and certifications to provide variety. For example, replace ”Python” with ”R” or ”SQL,” or add modern tools and technologies (e.g., ”Kubernetes,” ”Tableau,” or ”DataOps”).
- **Introduce Substantial Variations:** Adjust career progression to reflect diverse experiences (e.g., mix large enterprises, startups, freelance consulting, or non-profit work within the domain). Provide a mix of leadership, technical, and cross-functional roles depending on the job title provided. Rewrite responsibilities and achievements with unique metrics (e.g., ”Reduced costs by 20% through process automation” vs. ”Implemented a cloud-native solution, improving uptime by 30%”). Use alternate phrasing for similar contributions to reflect individual approaches and nuances. Change project goals or contexts while maintaining similar skills and challenges (e.g., instead of ”redesigning a retail app,” describe ”modernizing a fintech dashboard”). Introduce realistic but varied timelines and experiences (e.g., different durations in roles, sabbaticals, or overlapping responsibilities).

Output Format: The resume should be produced in the following JSON format: example_json

Table 1: The prompts used in our methodology for the synthetic data generation.

Summarization Prompt
Priority Statement: The percentage weight indicates how much importance should be given to this section of the resume. Analyse the resumes according to this importance. **Task:** Summarize how each resume fulfills the requirements of this section and what each summary lacks in that section. Give a score to each section on how much it satisfies the condition in the job description. The score for each section should be out of 100. For each resume, along with the explanation about how it matches and lacks in each section, there should be a final dictionary containing job_title,skills_weight, experience_weight, education_weight, and achievement weight. Given the following job requirements and retrieved resumes, generate a summary based on weighted importance. **Job Details:** {job_description} **Resumes:** {matched_resumes} Provide a summary of why each CV matches the job description or what they lack from the job description. Keep the summaries very short and to the point. Write how the resume matches the job description and how it lacks anything that is required in the job description. Finally, give the ID of the resume that you think is best suited for the job. Only perform the task asked, nothing else.

Table 2: The prompts used in our methodology for summarizing the candidate Profiles.

Figure 3 shows that the BERTScore is generally high (above 0.7), indicating strong semantic similarity between synthetic and anchor resumes, while the relatively low ROUGE score suggests limited n-gram overlap. Additionally, the fluctuating yet consistently high 1-Jaccard similarity demonstrates substantial diversity among the synthetic resumes. These results indicate that, with this prompt, it is challenging to achieve the right balance between accurately representing real-world CVs—reflected in high ROUGE overlap—and maintaining diversity among the generated resumes.

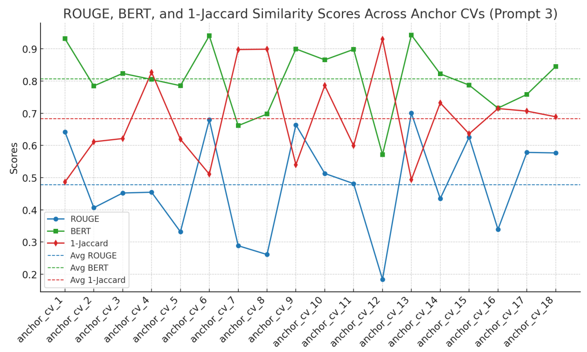

Figure 4: Results for the Synthetic Data Generation with Prompt 3

In figure 4, we can see the **Prompt 3** with one-shot setting results in higher BERT scores, showing that the synthetic resumes are closer semantically as well as having high word overlap represented by higher ROUGE score. Although a higher ROUGE score is desirable, the high ROUGE score here represents overfitting with the anchor CV. For that matter, we try to loosen the restriction with the **Prompt 4**.

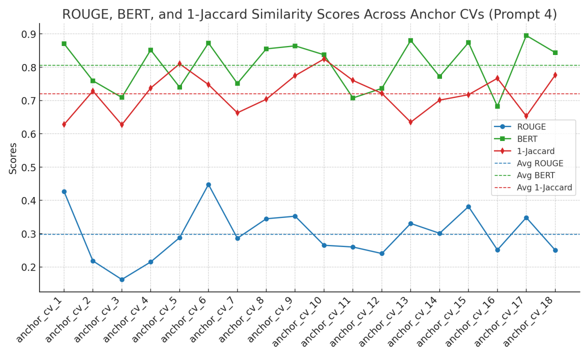

Figure 5: Results for the Synthetic Data Generation with Prompt 4

In Figure 5, the average ROUGE score decreases while the BERTScore remains comparable to that of Prompt 3. This indicates that the synthetic resumes retain strong semantic similarity to the anchor resumes without excessive word-for-word overlap, as evidenced by the lower ROUGE values. These results suggest that, with effective prompt engineering in a one-shot setting, it is possible to generate synthetic resume data that closely matches the real-world distribution while maintaining appropriate diversity.

4.2 RAG analysis:

4.2.1 Retriever:

When comparing the RAG retriever to the NER-based job-resume matching approach, the RAG retriever consistently outperforms the baseline across all evaluation metrics, as shown in Table 3. Notably, the accuracy metric remains consistent throughout, reflecting identical top-match correctness across both methods.

Method	Top-K	Precision@K	Recall@K	MRR	Accuracy
RAG	3	0.6667	0.6667	0.6500	0.6
NER	3	0.2167	0.2167	0.1833	0.1
RAG	5	0.8333	0.8333	0.6950	0.6
NER	5	0.3167	0.3167	0.2083	0.1
RAG	7	0.9000	0.9000	0.6950	0.6
NER	7	0.3167	0.3167	0.2083	0.1

Table 3: Comparison of RAG-based Retrieval vs. NER-based Matching

4.2.2 Generator:

The generator's performance in selecting the best resume was lower than that of the retriever, with accuracy dropping to 0.55 for (k=3). This decrease appears to stem from the nuanced differences between similar job roles, such as Data Scientist versus AI Scientist, and the overlapping skills and experiences found in positions like Senior Software Developer and Full Stack Developer. These subtle distinctions often led to ambiguity in the generator's predictions, contributing to the observed reduction in accuracy.

4.2.3 Human Evaluation:

Three human domain experts annotated the generated summaries using four criteria: relevance, completeness, clarity and conciseness, and accuracy. The results, presented in Figure 6, show consistent agreement among annotators on the relevance dimension, indicating that summaries generally captured key information and aligned resumes with job descriptions while highlighting missing criteria. For completeness, annotators 2 and 3 were more stringent, suggesting that some summaries lacked certain details. High scores and agreement in clarity and conciseness confirm that the summaries were mostly concise, well-structured, and informative. However, there was disagreement regarding accuracy, with annotator 2 noting several cases of misinformation or misinterpretation by the LLM—for example, incorrectly marking an incomplete PhD as finished, or failing to distinguish between "Ms. Data Science" and "Ms. Informatics" when interpreting job requirements.

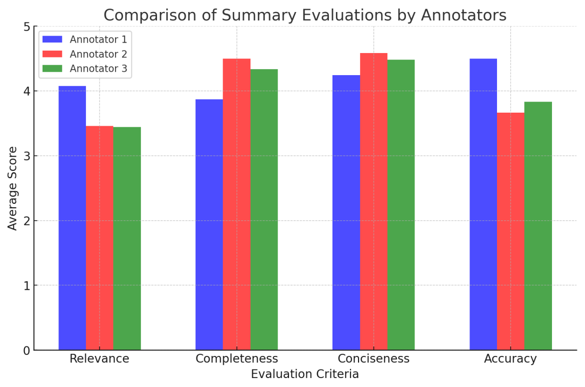

Figure 6: Result of the domain expert evaluation of Generator Summaries

5 Conclusion

Our experiments demonstrate that with careful prompt engineering, it is possible to generate high-quality, diverse synthetic resume datasets using models like `Mixtral-8x22B-Instruct-v0.1`, which balance semantic similarity with real-world resumes while ensuring diversity among outputs. We showed that RAG-based resume evaluation systems can be effectively assessed using both traditional retrieval metrics (such as Precision@k, Recall@k, Accuracy, and MRR) and detailed human evaluations by domain experts, revealing challenges like misinterpretation of qualifications. Future research could expand the dataset to diverse industries and job roles, incorporate advanced prompt engineering and fact verification, integrate domain-specific embeddings or knowledge graphs for better retrieval and ranking, and benchmark additional LLMs, all of which would help create more robust and reliable AI-driven hiring tools.

References

Meta AI. 2024a. Introducing meta llama 3.1: Advancing open-source ai.

Mistral AI. 2024b. Mixtral 8x22b model. https://mistral.ai.

André Bauer, Simon Trapp, Michael Stenger, Robert Leppich, Samuel Kounev, Mark Leznik, Kyle Chard, and Ian Foster. 2024. Comprehensive exploration of synthetic data generation: A survey.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Ashish Virendra Chandak, Hardik Pandey, Gourav Rushiya, and Harsh Sharma. 2024. Resume parser and job recommendation system using machine learning. In *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)*, pages 157–162.

Jenny Chim, Julia Ive, and Maria Liakata. 2025. Evaluating synthetic data generation from user generated text. *Computational Linguistics*, 51(1):191–233.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2025. The faiss library.

Chengguang Gan, Qinghao Zhang, and Tatsunori Mori. 2024. Application of llm agents in recruitment: A novel framework for resume screening.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing.

Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37:547–579.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10443–10461, Singapore. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Frank P. W. Lo, Jianing Qiu, Zeyu Wang, Haibao Yu, Yeming Chen, Gao Zhang, and Benny Lo. 2025. Ai hiring with llms: A context-aware and explainable multi-agent framework for resume screening.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On LLMs-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082, Bangkok, Thailand. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Öykü Mercan, Sena Cavsak, Aysu Deliahmetoglu, and Senem Tanberk. 2023. Abstractive text summarization for resumes with cutting edge nlp transformers and lstm. pages 1–6.

George A Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Mihai Nadăș, Laura Dioșan, and Andreea Tomescu. 2025. Synthetic data generation using large language models: Advances in text and code. *IEEE Access*, page 1–1.

OpenAI. 2024. Gpt-4o-mini.

Naoki Otani, Nikita Bhutani, and Estevam Hruschka. 2025. Natural language processing for human resources: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 583–597, Albuquerque, New Mexico. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Panagiotis Skondras, Panagiotis Zervas, and Giannis Tzimas. 2023. Generating synthetic resume data with large language models for enhanced job description classification. *Future Internet*, 15(11).

Thatavarthi Giri Sougandh, Sai Snehith K, Nithish Sagar Reddy, and Meena Belwal. 2023. Automated resume parsing: A natural language processing approach. In

2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), pages 1–6.

Akshata Upadhye. 2024. A comprehensive study of resume summarization using large language models. *International Journal of Computer Applications*, 186(6):33–37.

S.P Warusawithana, N.N. Perera, R.L. Weerasinghe, T.M. Hindakaraldeniya, and G. Ganegoda. 2023. Layout aware resume parsing using nlp and rule-based techniques. pages 1–5.

Mohamed T. Younes, Omar Walid, Mai Hassan, and Ali Hamdi. 2025. Mlar: Multi-layer large language model-based robotic process automation applicant tracking.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.