Beyond Human Labels: A Multi-Linguistic Auto-Generated Benchmark for Evaluating Large Language Models on Resume Parsing

Zijian Ling[1]* , Han Zhang[1]* , Jiahao Cui[1] , Zhequn Wu[1] ,
Xu Sun[2] , Guohao Li[3,4] , Xiangjian He[2]†
[1]Apply U UK, [2]University of Nottingham Ningbo, China, [3]Eigent.AI , [4]CAMEL-AI.org

Abstract

Efficient resume parsing is critical for global hiring, yet the absence of dedicated benchmarks for evaluating large language models (LLMs) on multilingual, structure-rich resumes hinders progress. To address this, we introduce ResumeBench, the first privacy-compliant benchmark comprising 2,500 synthetic resumes spanning 50 templates, 30 career fields, and 5 languages. These resumes are generated through a human-in-the-loop pipeline that prioritizes realism, diversity, and privacy compliance, which are validated against real-world resumes. This paper evaluates 24 state-of-the-art LLMs on ResumeBench, revealing substantial variations in handling resume complexities. Specifically, top-performing models like GPT-4o exhibit challenges in cross-lingual structural alignment while smaller models show inconsistent scaling effects. Code-specialized LLMs underperform relative to generalists, while JSON outputs enhance schema compliance but fail to address semantic ambiguities. Our findings underscore the necessity for domain-specific optimization and hybrid training strategies to enhance structural and contextual reasoning in LLMs.

1 Introduction

In today's competitive global job market, efficient parsing and interpretation of resumes have become critical for recruiters and organizations (Jalili et al., 2024). Nearly 70% of companies use online recruiting platforms (Paramita, 2020). Traditional resume parsing approaches, relying on rule-based systems or machine learning models trained on structured datasets within a single language or cultural context (Tallapragada et al., 2023). Challenges include scarcity of high-quality training data due to privacy concerns and regulations like the General Data Protection Regulation (GDPR) (Javed et al., 2015), the lack of standardized resume templates leading to variability in format and content (Sachan et al., 2024; Chen, 2022).

Large language models (LLMs) have revolutionized structured document processing, achieving high performance in table extraction (Chen, 2022), multilingual form understanding (Wibawa et al., 2024), and synthetic data generation for sensitive domains like healthcare (Gu et al., 2024). However, their application to resume parsing remains unexplored, which may due to: structural heterogeneity (mixing tables, lists, and free text), cross-lingual complexity, and privacy constraints that limit real-world data availability, as shown in Figure1. Current benchmarks focus on homogeneous formats (e.g., JSON (Ojokoh and Adebisi, 2018)) or simplistic synthetic resumes (van Breugel and van der Schaar, 2024), failing to reflect the complexity of real-world applications. This gap leaves LLMs' ability to parse hybrid layouts and infer implicit resume semantics (e.g., skill relevance to job roles) unverified.

Hence, we present **ResumeBench**, the first multilingual benchmark comprising **2500** synthetic resumes spanning **50 templates** and **5 languages**, generated via a human-in-the-loop pipeline to ensure cross-lingual realism and alignment with real-world resumes. To ensure that our synthetic data capture the full complexity of real-world situations, we further collect authentic resumes to build the ResumeBench-Mix dataset. Our evaluation of **24** LLMs on ResumeBench, both proprietary and open-source, reveals key limitations. While larger models generally perform better, domain-specific optimization remains crucial. Even advanced models like GPT-4o struggle with parsing hybrid layouts, such as multilingual resume formats. Notably, enabling JSON mode improves schema adherence for larger models, yet semantic ambiguities per-

*Zijian Ling and Han Zhang contributed equally to this work (co-first authors).
†Corresponding author. Email: Sean.He@nottingham.edu.cn.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 31908–31934
November 4-9, 2025

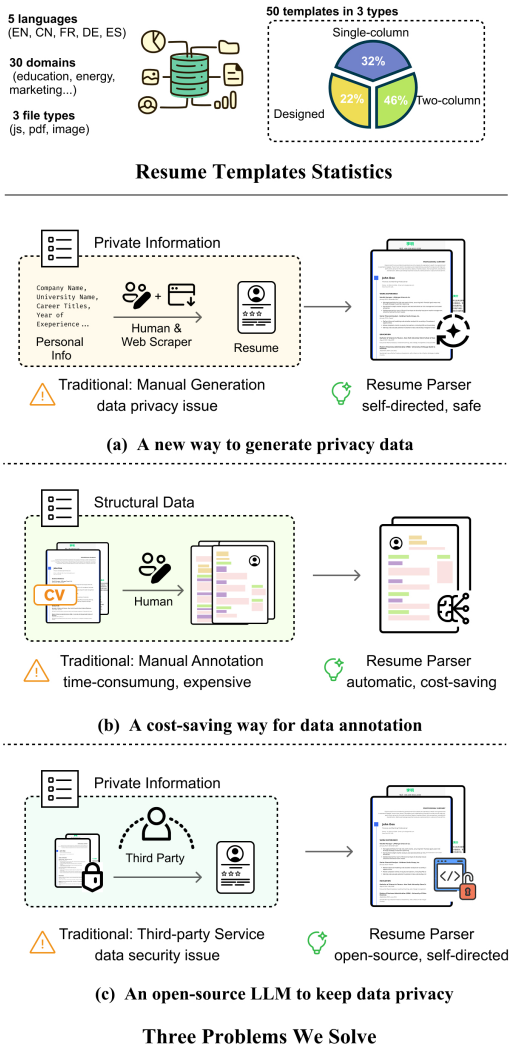


Figure 1: Overview of Challenges and Solutions in Resume Parsing

sist, highlighting unresolved challenges in implicit skill inference. By bridging the gap between synthetic data fidelity and real-world complexity, ResumeBench establishes an auto-generated framework for advancing multilingual structured document processing. It offers actionable insights into parsing robustness, structural accuracy, and cross-lingual reliability, supporting large language model adaptation and auditing in recruitment technology.

2 Related Work

2.1 LLMs in Structure-rich Text Understanding

Early LLMs like BERT and GPT-2 excelled in producing fluent text but often struggled to adhere to specific schemas or formats necessary for generating structure-aware tasks (Devlin et al., 2018; Radford et al., 2019). Chain-of-thought (CoT) prompting improved performance on tasks like legal parsing and table extraction by breaking tasks into intermediate reasoning steps (Wei et al., 2023; Hazourli, 2022). However, CoT faces limitations when applied to implicit structural dependencies such as skill hierarchies in resumes or fragmented educational timelines.

Progress has been made with benchmarks like TableBench (Wu et al., 2025)and StructLM (Zhuang et al., 2024), which handle tabular data well but struggle with global schema inference. Current benchmarks also fall short in capturing the linguistic and cultural diversity inherent in real-world resumes, which often feature region-specific layouts. Tool-augmented LLM, for example, Toolformer (Schick et al., 2023), enhances reasoning by integrating external tools, but the application to resumes with diverse formats remains unexplored.

The lack of publicly available, diverse annotated datasets further complicates the development of generalizable LLM models for resume parsing. Current datasets tend to be monolingual or overly simplified, failing to capture the complexities and variations inherent in global resumes (Ingvar et al., 2021).

2.2 LLM-Enabled Synthetic Data Generation

Recent advances in LLMs position synthetic data generation as a scalable solution for privacy-constrained domains (Long et al., 2024). LLMs enable conditional generation using prompts with structural constraints (Eldan and Li, 2023; Zhuang et al., 2024). Human-in-the-loop validation improves quality and diversity, ensuring alignment with career-specific attributes, including job titles and skill terminologies (Li, 2017; Gu et al., 2024).

Despite progress, challenges persist. Balancing realism and fidelity in prompts is critical—over-constraint leads to repetitive outputs, while under-constraint introduces noise (Liu et al., 2024). Bias mitigation, such as addressing gender disparities in job titles, requires debiasing prompts and diversity-aware sampling strategies (Wilson and Caliskan, 2024). Finally, cross-lingual consistency demands schema alignment across languages, a challenge also observed in multilingual form parsing (Wibawa et al., 2024). Addressing these issues is critical for realizing the full potential of LLM-based synthetic data generation in real-world settings.

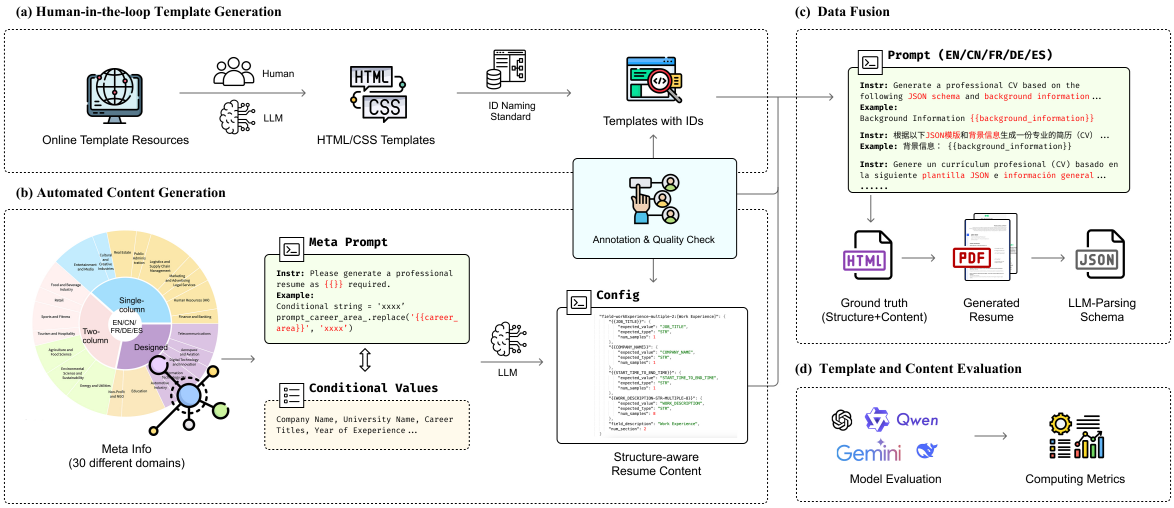


Figure 2: The Pipeline of Data Generation And Evaluation

3 ResumeBench

ResumeBench is designed to convert raw PDF resumes and corresponding parsing prompts into structured JSON format. The dataset follows a predefined schema that ensures consistency across diverse resume formats. The pipeline of data generation consists of three stages as shown in Figure 2. First, a human-in-the-loop process creates diverse resume templates with corresponding annotation labels. Next, LLMs automate content generation, producing structure-aware resumes in multiple languages, including English, Chinese, German, French and Spanish. Finally, the generated content is integrated with the templates and undergoes a thorough quality and accuracy review.

3.1 Human-in-the-loop Pipeline for Template Generation

Our template generation employs two primary approaches as follows.

- **Collection from multiple sources:** We collect resume templates from different online platforms, covering layouts like single-column, double-columns, and top-down designs. These templates are then converted into HTML and CSS, focusing only on their structure and visual appearance, without any interactive features.

- **Utilization of structure-rich text templates:** We also leverage LaTeX templates, converting them into HTML and CSS using a semi-automatic batch process facilitated by ChatGPT (GPT-4o) (OpenAI, 2022, 2024).

Subsequently, we manually assign unique ID attributes to selected template tags, enabling automated structure-aware operations with seamless integration and consistency in content generation and data fusion.

Human involvement, crucial for ensuring reliable data (Li, 2017), is incorporated to review the outputs, ensuring diverse structures and layouts while avoiding duplication. A total of 50 templates are finalized in this phase.

3.2 Automated Resume Generation

3.2.1 Conditional Attributes Generation

To ensure diverse resume content, we adopt the LLM-based generation paradigm Yu et al. (2024), using attributed prompts to create varied resumes across career paths. Unlike prior methods (Bruera et al., 2022), which sample attributes from existing datasets, we generate them directly using ChatGPT (GPT-4o) (OpenAI, 2022, 2024). This enables career domain generation in multiple languages, enhancing cultural diversity. Human reviewers validate realism and domain adherence, while attributes such as universities, companies, and job titles are conditionally generated for consistency. This approach ensures diverse yet realistic resumes while maintaining quality standards.

3.2.2 Configuration

We define a configuration for each resume template, outlining the content structure across differ-

ent sections. This enables dynamic variation in the number and arrangement of elements, such as work experience and education.

The templates consist of non-nested and nested blocks. Non-nested blocks cover simple sections like personal details, while nested blocks manage complex sections, allowing flexible repetition of sub-blocks. We assign unique ID value to each block to guide the LLMs in content generation. Additionally, for nested blocks, dynamic identifiers are introduced to enable flexible content generation, enhancing structural diversity and layout variability.

3.2.3 Structure-Aware Content Generation

An essential step in generating synthetic resumes using LLMs is ensuring that the content adheres to a specific resume format. We convert the HTML structure of each resume template into a configuration file that stores both structural and semantic information necessary for generating content. To facilitate structured content generation, we utilize JSON format, which is widely supported by modern LLMs. In particular, we leverage GPT-4o's Structured Generation[1] to ensure precision and consistency in the content generation process. For each template, we randomly sample multiple sets of conditional attributes, allowing for controlled variability across resumes. This approach ensures that the generated content remains diverse while conforming to the structural and thematic expectations defined in the template.

3.2.4 Data Fusion

The final step in the content generation process involves merging the generated content with the corresponding resume templates and producing the final output in PDF format. This ensures that the synthesized resumes are ready for practical use and evaluation in NLP tasks. We integrate the generated content back into the HTML templates, producing fully formatted resumes. These HTML files are subsequently converted into PDF documents using pdfkit[2], resulting in a diverse dataset of 2,500 resumes (see samples in Appendix P).

3.3 Dataset Statistic

Resume Templates As illustrated in Figure 1, our dataset comprises a total of 50 resume templates, categorized by layout styles: Double-columns[3], Single-column[4] templates and more stylistically complex formats named designed[5] format.

Dataset	Domains	Parsing Annotation
Jiechieu and Tsopze (2021)	1	×
Resume Dataset on Kaggle[6]	24	×
ResumeBench	30	✓

Table 1: Comparison with Public Resume Datasets

Resume Domains There are very few publicly available resume datasets designed for NLP tasks. We compare our work with two accessible datasets: the real-world English resume dataset (Jiechieu and Tsopze, 2021), which focuses on engineering-related resumes, and a broad-spectrum career dataset[6] (Wilson and Caliskan, 2024). In comparison with these existing datasets, ResumeBench covers a wider range of career domains, as shown in Table 1.

3.4 Integration of Real-World Resume Data

To demonstrate the effectiveness of our framework and benchmark, we collect real, public resume samples in five languages: English, Chinese, Spanish, French, and German, from various professional domains using internet sources (e.g., Google Images). This dataset is referred to as Resumebench-Real. Additionally, we sample an equal number of synthetic resume samples from Resumebench in the same five languages to create a combined real-synthetic dataset, which we name Resumebench-Mix (Appendix Q).

3.5 Dataset Analysis

To demonstrate the diversity and truthfulness of the proposed dataset, we employ widely used common metrics for machine generated text (see Appendix O for details) as well as using a LLM-as-a-judge way to validate the dataset with 20% of samples in Resumebench-Mix. We take the follow-

[1]Available at: https://openai.com/index/introducing-structured-outputs-in-the-api/

[2]Available at: https://github.com/JazzCore/python-pdfkit

[3]A structured layout with two columns. One typically contains personal details like contact information and a summary, while the other presents professional experience and achievements.

[4]A straightforward format where experiences and information are listed sequentially from top to bottom.

[5]A flexible template without a fixed structure, allowing experiences to be customized and positioned in different sections as needed.

[6]A kaggle resume dataset available at https://www.kaggle.com/datasets/snehaanbhawal/resume-dataset

Model	Type	SR ↑		KM Ratio ↑		TED ↓		ROUGE-L ↑		BERTScore ↑	
		w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON
Proprietary Models											
GPT-4o	API	0.9912	0.9816	0.5963	0.6020	68.45	67.51	0.6953	0.6908	0.8925	0.8972
GPT-4o-mini	API	0.9936	0.9908	0.5674	0.5636	56.58	56.36	0.6882	0.6930	0.9071	0.9087
Gemini-2.0-Flash	API	0.8840	0.8680	0.6100	0.6261	74.42	76.35	0.6869	0.7034	0.9000	0.9034
Qwen-2.5-0.5B	Local	0.9136	0.6756	0.1702	0.1355	113.29	112.76	0.3928	0.4173	0.7813	0.7933
Llama-3.2-1B	Local	0.8716	0.4640	0.4464	0.4703	90.15	89.51	0.4487	0.4498	0.7959	0.7914
Qwen-2.5-1.5B	Local	0.8044	0.6528	0.1719	0.1869	94.05	88.92	0.5736	0.5935	0.8509	0.8582
Qwen-2.5-3B	Local	0.9812	0.9160	0.5235	0.5363	64.92	62.52	0.6335	0.6472	0.8732	0.8814
Llama-3.2-3B	Local	0.7732	0.3316	0.3182	0.4687	100.25	76.17	0.5551	0.6562	0.8339	0.8770
Qwen-2.5-7B	Local	0.9888	0.9396	0.6007	0.5934	65.92	65.58	0.6943	0.6957	0.8893	0.8908
Llama3.1-8B	Local	0.9496	0.9020	0.5873	0.5943	57.22	53.47	0.6895	0.7140	0.9068	0.9172
Falcon3-7B	Local	0.9480	0.9004	0.4211	0.4376	73.57	72.27	0.6852	0.6886	0.9017	0.9035
Qwen2.5-Coder-7B	Local	0.9828	0.9544	0.3530	0.3548	78.39	78.31	0.5922	0.5758	0.8577	0.8522
CodeLlama-7B	Local	0.9124	0.8992	0.3903	0.3444	90.72	93.19	0.4242	0.3284	0.7556	0.7068
Qwen-2.5-14B	Local	0.9932	0.9484	0.5823	0.5834	65.50	63.35	0.7111	0.7334	0.9226	0.9315
Qwen-2.5-32B	Local	0.9968	0.9628	0.5065	0.4996	64.76	62.97	0.6809	0.7003	0.8854	0.9119
CodeLlama-34B	Local	0.9552	0.6536	0.5197	0.5030	76.08	69.78	0.6710	0.6801	0.8976	0.9063
Qwen2.5-Coder-32B	Local	0.9972	0.9596	0.6117	0.6203	66.44	66.03	0.7030	0.7098	0.8962	0.9030
Qwen-2.5-72B	Local	0.9972	0.9612	0.6158	0.6156	61.33	57.67	0.7128	0.7351	0.9082	0.9280
Llama3.1-70B	Local	0.9708	0.958	0.5633	0.5515	63.83	63.39	0.7283	0.7328	0.9207	0.9271
Ministral-8B-2410	Local	0.9004	0.8292	0.3054	0.3109	75.26	74.28	0.6214	0.6189	0.8799	0.8771
Mistral-Small-24B-2501	Local	0.9924	0.9580	0.5886	0.5920	65.31	64.30	0.6958	0.7073	0.8898	0.9017
Reasoning Models											
DeepSeek-R1-Distill-Qwen-1.5B	Local	0.7236	0.5824	0.4188	0.4202	106.26	96.10	0.3789	0.3974	0.7632	0.7795
DeepSeek-R1-Distill-Qwen-7B	Local	0.8964	0.8096	0.4929	0.4990	73.48	75.64	0.6397	0.6243	0.8812	0.8754
DeepSeek-R1-Distill-Llama-8B	Local	0.7800	0.9448	0.5344	0.5645	65.68	62.82	0.6666	0.6749	0.8917	0.8995

Table 2: Performance of LLMs on ResumeBench. The Metrics (KM Ratio, TED, ROUGE-L, BERTScore) Are Computed as The Average Score across All Success Samples. We present the average scores across all samples in Appendix A, while the SR in both tables is calculated based on all samples.

ing prompt to guide ChatGPT[7]

```
Determine if the given file is a real or synthetic resume. Ignore photo, name, email, and other personal information, as these have been pre-processed. Provide your decision only, without any additional explanation
```

to make binary classification on both real and synthetic samples. As show in Figure 6, ChatGPT is 100% confident in real samples classification but 56% of our synthetic samples can pass the judge.

4 Experiments

4.1 Experiment Setup

We conduct a comprehensive evaluation of 24 LLMs, including both general and code-focused models, as well as proprietary and open-source models. For proprietary LLMs, our evaluation includes GPT-4o and GPT-4o-mini (Achiam et al., 2023) from OpenAI, Gemini-2-Flush from Google (Google, 2024). For open-source models, we assess models from Llama3.1 (Dubey et al., 2024), Qwen2.5 (Yang et al., 2024), Mistral (AI, 2025), CodeLlama (Roziere et al., 2023), Qwen2.5-Coder (Hui et al., 2024), Falcon (Team, 2024) and DeepSeek-R1-Distill models (DeepSeek-AI et al., 2025). All selected open-source models are official instruct or chat versions, with model sizes ranging from 0.5B to 72B, chosen to accommodate GPU memory constraints and represent the latest available versions.

To convert resume PDFs for input into the LLMs, we utilize the pypdf[8] package to extract text from the PDF files. All LLMs are evaluated in a zero-shot setting, using only a user prompt aligned with the corresponding parsing schema. Additionally, we also conducted few-shot prompting ablations (Appendix B). We observed that adding 1–3 in-context examples can improve semantic metrics (e.g., ROUGE-L/BERTScore) and sometimes strengthen structural alignment (KM Ratio/TED) on successful parses, but it often reduces the overall success rate due to more invalid or partially valid outputs. Given this trade-off, we retain zero-shot as the main-text setting for a fair and consistent baseline, while reporting the complementary few-shot gains and their limitations in Appendix B.

4.2 Evaluation Setup

Resumes typically consist of two key components: the resume template and the resume content, each presenting unique challenges. To ensure a thorough and comprehensive evaluation, we propose distinct evaluation strategies for each component.

The evaluation task is conducted based on a parsing schema prompt provided to a LLM, where p represents the **Parsing Schema Prompt** (See

[7]We use chatgpt-4o-latest https://platform.openai.com/docs/models/continuous-model-upgrades

[8]Available from: https://github.com/py-pdf/pypdf

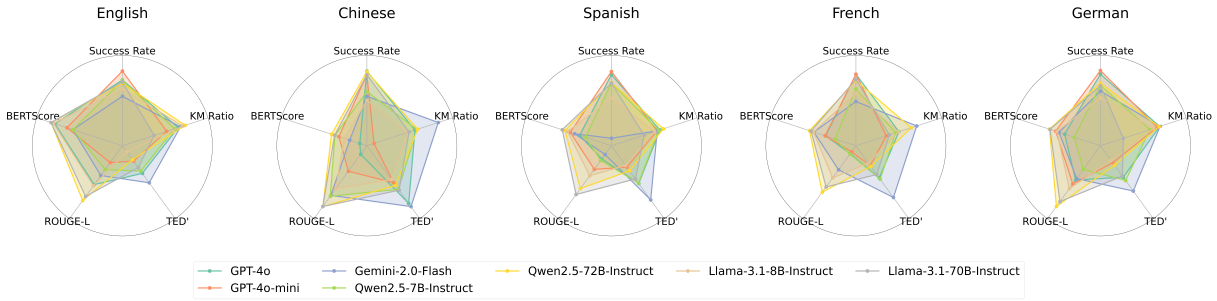


Figure 3: Performance of LLMs Across Five Languages in ResumeBench. KM Ratio, TED, ROUGE-L, and BERTScore are computed as the average scores across all success samples. SR is calculated based on all samples. We apply Min-Max scaling to transform the scores into a more visually interpretable range, scaling them between 0.1 and 0.9.

Appendix N), r represents the **PDF File** containing the resume, o represents the **Output** from the LLM, and gt represents the **Ground Truth**, i.e., $GT = \{GT_{\text{template}}, GT_{\text{content}}\}$. Then, the output is

$$o = \text{LLM}(p, r). \tag{1}$$

4.3 Resume Template Evaluation

To evaluate the resume template, we employ three complementary approaches, each providing a distinct perspective on the structural aspects of the resume. We conceptualize the resume structure as a hierarchical tree, where each section name serves as a node, and edges recursively link each section name to its corresponding child subsections.

First, we calculate the Success Rate (SR) for the success of converting the outputs of LLMs to valid JSON formats using Equation 3.

$$f_n(o_i, d_i) = \begin{cases} 1 & \text{if conversion is successful,} \\ 0 & \text{if conversion fails,} \end{cases} \tag{2}$$

where f_n is the output conversion operation, o_i is the output of the LLM conditioned on d_i such that $d_i \supset [p_i, r_i]$ from dataset D_j, and $d_i \in D_j$. Additionally, $D_j \in \{D_{\text{en}}, D_{\text{zh}}\}$ denotes the dataset being either D_{en} (English) or D_{zh} (Chinese).

$$\text{SR}(\text{LLM}_i, D_j) = \frac{\sum_{d_i \in D_j} f_n(o_i, d_i)}{|D_j|}. \tag{3}$$

Second, we adopt a Layer-Wise Key Matching Ratio (KM Ratio) to assess the structural alignment of the generated resumes with the expected template, see Equation 5. For a single resume r, the Layer-Wise Key Matching Ratio (KM Ratio) is defined by:

$$\text{KM Ratio}(r) = \frac{1}{L_r} \sum_{l=1}^{L_r} \frac{|S_{l,r} \cap G_{l,r}|}{|S_{l,r}|}, \tag{4}$$

where L_r is the number of layers in the hierarchical tree for the resume r, $S_{l,r}$ is the set of expected section names, and $G_{l,r}$ is the set of generated section names at layer l for document r.

We report the average KM Ratio over the dataset with N resumes, we compute:

$$\text{Average KM Ratio} = \frac{1}{N} \sum_{r=1}^{N} \text{KM Ratio}(r). \tag{5}$$

Third, to enhance structural awareness in our evaluation, we employ the Tree Edit Distance (TED) metric (Zhang, 1996). This method allows for a direct comparison of the hierarchical structures of keys in the parsing output JSON and the ground truth, facilitating a more nuanced assessment of the overall structural arrangement of the resume.

Given the nature of JSON data structures, which allow for unordered sibling nodes, we specifically measure the unordered TED (Paaßen, 2018). This approach focuses solely on ancestor relationships, making it well-suited for our task.

The unordered TED between the parsed result t_1 and the ground truth t_2 is defined by:

$$TED(t_1, t_2) = \min_{S} \{\gamma(S) \mid S(t_1 \text{ into } t_2)\}, \tag{6}$$

where S represents the set of edit operations, and $\gamma(S)$ is the cost function assigned to each operation.

We report the average TED over a dataset containing N resumes by:

$$\text{Average TED} = \frac{1}{N} \sum_{i=1}^{N} TED(t_{1,i}, t_{2,i}), \tag{7}$$

where $t_{1,i}$ represents the ground truth for the i-th resume, and $t_{2,i}$ is the corresponding parsed result.

4.4 Resume Content Evaluation

For the evaluation of resume content, we utilize ROUGE-L (Lin, 2004) to assess the quality of the parsed resume outputs. ROUGE-L focuses on the longest common subsequence (LCS) between the generated text and the ground truth. Additionally, we employ BERTScore Zhang et al. (2020) to evaluate the semantic similarity between the generated content and the reference data. To calculate the average of the both metrics over the dataset, we exclude any structural keys from the parsed results and the ground truth, focusing on the resume content.

5 Discussion

5.1 Model Performances on ResumenBench

Our experiments provide critical insights into the performance of LLMs on resume parsing tasks (see Table 2). **While model scale generally correlates with parsing accuracy, we observe exceptions**. For instance, Llama3.2-3B underperforms Llama3.2-1B in SR (77.32% vs. 87.16%), despite its smaller size. Larger models like Qwen2.5-72B show marginal improvements over Qwen2.5-7B in SR (99.72% vs. 98.88%, +0.84%) and BERTScore (90.82% vs. 88.93%, +2.13%), although there are larger discrepancies in KM Ratio (+1.51%) and TED improvement (-6.96%).

There is no clear improvement in coding-specific models over their general instruction-tuned counterparts at the same parameter scale (e.g., Qwen2.5-Coder-7B vs. Qwen2.5-7B and Qwen2.5-Coder-32B vs. Qwen2.5-32B). In fact, the coding-specific models show lower performance in TED. For example, Qwen2.5-Coder-7B, underperform generalist counterparts in TED by 18.9% (78.39% vs. 65.92%), while Qwen2.5-Coder-32B shows a smaller TED gap (+2.6% vs. Qwen2.5-32B).

Reasoning models consistently underperform compared with their base instruct models across all tested parameter sizes (from 1.5B to 8B), with a particularly notable drop in the SR. Reasoning models like DeepSeek-R1-Distill-Qwen-1.5B exhibit a 10.1% drop in SR compared to base models, underscoring the need for hybrid training strategies. This drop, particularly in the reasoning models, emphasizes the need for hybrid training objectives that add schema learning (via techniques like JSON mode) to improve logical inference within LLM frameworks.

Proprietary models remain strong, yet open-source models demonstrate competitive performance. Open-source models, such as Qwen2.5-70B, achieve top scores in SR and KM Ratio with JSON mode, while Qwen2.5-14B leads in BERTScore, underscoring their competitiveness in structured prediction.

5.2 Model Performances on Resumebench-Mix

Synthetic data enables strong performance in our evaluation (see Table 3). Specifically, Qwen-2.5-7B achieves near-perfect SR (99.05%) on synthetic resumes vs. 96.21% on real data (-2.84% drop). Llama3.1-8B, however, improves on synthetic data (83.89% SR vs. 82.46% real, +1.43%), contradicting earlier claims of degradation. The high KM Ratio (Qwen: 0.8844, Llama: 0.872) and BERTScore (Qwen: 0.9109, Llama: 0.9657) on synthetic data confirm structural and semantic fidelity.

For all ablation experiments, we adopt parsing schemas distinct from those used in Resumebench. (see Appendix M).

5.3 How do LLMs perform on multi-lingual resumes?

Across multiple model variants (Qwen2.5-7B/72B-Instruct and Llama-3.1-8B/70B-Instruct), **we observe significant cross-linguistic disparities in resume parsing**, as reflected in key evaluation metrics such as SR, KM Ratio, and TED, in Figure 3. English consistently demonstrates the highest SR and robust structural alignment, which may be attributed to its syntactic regularity and predominant representation in pretraining corpora. Chinese occasionally achieves comparable SR, but it exhibits higher TED values, suggesting challenges related to segmentation and distinct structural conventions. Spanish and French present intermediate performance, potentially constrained by morphological complexity and locale-specific formatting variations. In contrast, German demonstrates comparatively robust performance, possibly benefiting from its frequent inclusion in multilingual benchmarks. Moreover, cross-lingual generalization remains inconsistent across the evaluated models, highlighting the need for enhanced training methodologies and more comprehensive evaluation strategies, particularly to improve performance in low-resource language scenarios.

[1]TED is scale by $log(v, 100)$, where v is variable and 100 is the log base.

Model	Type	SR		KM Ratio ↑		TED ↓		ROUGE-L ↑		BERTScore ↑	
		Real	Synthetic	Real	Synthetic	Real	Synthetic	Real	Synthetic	Real	Synthetic
Qwen-2.5-7B	Local	0.9621	0.9905	0.7869	0.8844	63.85	31.52	0.7665	0.8607	0.8950	0.9109
Llama3.1-8B	Local	0.8246	0.8389	0.7299	0.872	31.49	18.90	0.7653	0.9272	0.8812	0.9657

Table 3: LLMs Performance on ResumeBench-Mix

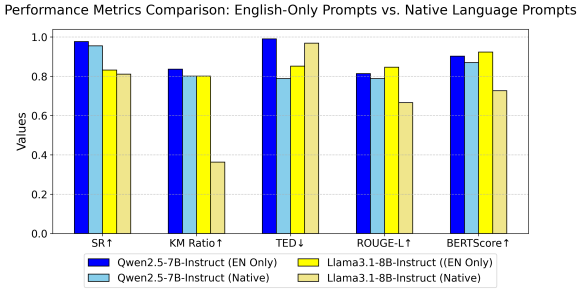


Figure 4: Performance Metrics Comparison: English Only Prompts Vs. Native Language Prompts.[1]

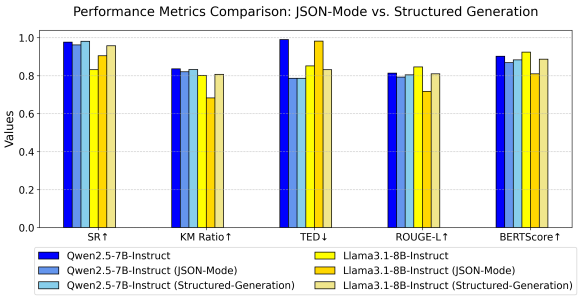


Figure 5: Performance Metrics Comparison: No-Constraint Vs. JSON Mode Vs. Structured Generation.[1]

To estimate the impact of prompt language on model performance, we consider two ablation experimental settings. In the first setting, the prompts are presented exclusively in English, while in the second setting, the prompts are provided in the corresponding native language for each sample. Across the five metrics illustrated in Figure 4, with the exception of the TED score for the Qwen model, the results consistently indicate that the performance under the second setting is inferior compared to the English-only setting. This observation suggests that the **model demonstrates better performance when prompted in English, even when both the data and prompts are presented in another language**.

5.4 Does 'JSON Mode' improve the performance of LLMs?

Enabling JSON mode consistently improves structured data extraction across a range of LLMs, often boosting the SR by double-digit margins compared to free-text outputs. Larger open-source models like Qwen2.5-72B also excel in JSON mode (99.72% SR), while code-specialized models such as CodeLlama-34B show stark improvements (95.52% vs. 65.36% w/o JSON).

While JSON mode improves structural adherence, it does not resolve semantic ambiguities (e.g., Qwen2.5-7B's KM Ratio increases to 60.07% with JSON vs. 59.34% without). Semantic metrics like BERTScore remain largely unaffected (e.g., GPT-4o: 89.25% vs. 89.72% without JSON). Coding-specialized models, such as CodeLlama-7B, underperform generalist counterparts in nested field extraction (TED: 90.72 vs. 65.92 for Qwen2.5-7B), contradicting assumptions about their superiority.

In comparison, we explore the effectiveness of structured generation in Figure 5, contrasting it with JSON mode and no constraints. Both JSON mode and structured generation improve the SR, with structured generation providing greater benefits. For structure-related metrics (KM Ratio, TED), JSON mode decreases structural parsing performance, while structured generation slightly improves it. A similar trend is observed for semantic parsing, as shown by ROUGE-L and BERTScore. These results suggest that **JSON mode enhances structural accuracy but may weaken semantic performance, whereas structured generation offers a more balanced improvement, making it a better choice for resume parsing**.

5.5 Qualitative and Error Analysis

Beyond aggregate metrics, our evaluation incorporates qualitative case studies and detailed error categorization (Appendix E). For example, a resume with multiple nested roles in the Work Experience section was parsed differently across model scales. Large-capacity models (e.g., GPT-4, Qwen-2.5-72B) preserved the role hierarchy and bullet-level details, whereas smaller models (e.g., 7B scale) collapsed the layout—merging bullet points across roles and omitting one role-specific description. This structural loss reduced KM Ratio and increased TED, while omissions lowered ROUGE-L and BERTScore, illustrating the challenges of parsing deeply nested fields.

From such cases, we identify two main failure categories (Appendix E.3). The first, complete parsing failures, produce invalid JSON (e.g., missing brackets, empty outputs), directly lowering Success Rate. The second, partial parsing errors, yield syntactically valid JSON but with structural misalignment (e.g., incorrect nesting, cross-role conflation) or semantic omissions (e.g., truncated bullet points, missing sub-sections). The nested-role example exemplifies this latter type. Both error classes degrade evaluation metrics: structural errors reduce KM Ratio and increase TED, while omissions depress ROUGE-L and BERTScore. Notably, smaller models exhibited more partial errors in domains such as finance and public administration, where jargon density and complex layouts increase parsing difficulty.

Taken together, these analyses demonstrate that averages can obscure hard cases and emphasize the need to improve both JSON validity and hierarchical coherence for reliable resume parsing.

5.6 Additional Analysis

To inform deployment, our additional analyses reveal a few consistent patterns. First, lowering the sampling temperature improves validity and stability, making conservative decoding a safe default (Appendix C). Second, incorporating visual modality with VLMs better preserves layout structures that text-only pipelines often lose (Appendix D). Third, under structured outputs, large models reach ceiling success rates across languages while smaller models gain the most; English prompts remain the most reliable, though some non-English cases show segmentation or ordering issues (Appendix F–I, K). Fourth, template complexity matters: deeply nested single-column and asymmetric two-column formats degrade performance, motivating template-aware prompting or routing (Appendix J). Finally, efficiency scales predictably with size and constraints; mid-sized ($\sim$14B) models offer near–large-model accuracy with much lower latency, especially with optimized inference backends (Appendix L). Together, these findings support practical guidelines: conservative decoding, layout-sensitive modeling, and balanced model sizing for effective multilingual parsing.

6 Conclusion

We have presented ResumeBench, the first multilingual benchmark for evaluating LLMs on resume parsing, addressing structural heterogeneity, privacy constraints, and linguistic diversity. Our human-in-the-loop pipeline generates 2,500 synthetic resumes across 50 templates and 30 career domains, validated against real-world data. Comprehensive evaluation of 24 LLMs has revealed critical insights that, while proprietary models (e.g., GPT-4o) excel in schema adherence, cross-lingual structural alignment remains challenging. Smaller models exhibit inconsistent scaling, and code-specialized LLMs lag behind generalist counterparts. JSON outputs enhance syntactic fidelity but fail to resolve semantic ambiguities. ResumeBench bridges synthetic-to-real gaps in structured document understanding, offering a scalable framework for advancing multilingual parsing.

7 Ethical considerations

We recognize the risks of using LLMs for resume parsing, including biases in data generation, potential misuse in recruitment, and the possibility of generating fake resumes. To mitigate these, we adopted a human-in-the-loop process to ensure diversity and realism, alongside privacy- and ethics-aware dataset design. Debiasing was applied during data generation through unbiased prior knowledge and diversity-aware sampling. Clear usage guidelines further restrict malicious applications, such as synthetic data misuse. Our dataset comprises 2,500 synthetic resumes generated via a privacy-compliant pipeline, with synthetic names and details to avoid personally identifiable information. Sensitive fields (e.g., names, emails, contacts) were anonymized and manually reviewed to ensure privacy and remove offensive content. For ResumeBench-Mix, we relied solely on publicly available data and evaluated only on open-source models in a local setting, with no external exposure. This subset will not be publicly released unless specifically requested for research purposes and approved under privacy and ethical review. All artifacts introduced in this work, including RESUMEBENCH and RESUMEBENCH-MIX, are made available for non-commercial research and educational purposes under the CC BY-NC 4.0 license, consistent with the intended conditions of use.

8 Limitation

First, conducting a thorough failed case analysis and comparing model performance relative to parsing time could offer valuable insights into

the efficiency and effectiveness of different approaches. Second, language diversity, while addressed through the inclusion of five languages, does not encompass the full spectrum of linguistic and cultural variations found globally. The performance disparities observed across languages indicate potential model biases and underscore the need for more linguistically diverse training datasets. Future work could explore domain-adaptive training techniques and investigate the impact of language-specific prompting strategies more extensively.

9 Acknowledgment

The authors gratefully acknowledge Apply U UK, under whose affiliation this work was conducted.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Mistral AI. 2025. Mistral small 3. https://mistral.ai/en/news/mistral-small-3.

Andrea Bruera, Francesco Alda, and Francesco Di Cerbo. 2022. Generating realistic synthetic curricula vitae for machine learning applications under differential privacy. In *Proceedings of the Workshop on Ethical and Legal Issues in Human Language Technologies and Multilingual De-Identification of Sensitive Data In Language Resources within the 13th Language Resources and Evaluation Conference*, pages 53–63.

Dan Chen. 2022. *Artificial Intelligence (AI) in Employee Selection: How Algorithm-Based Decision Aids Influence Recruiters' Decision-Making in Resume Screening*. Ph.D. thesis, The University of Texas at Arlington.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Google. 2024. Gemini 2.0. https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024/?utm_source=deepmind.google&utm_medium=referral&utm_campaign=gdm&utm_content=.

Zhouhong Gu, Haoning Ye, Zeyang Zhou, Hongwei Feng, and Yanghua Xiao. 2024. Structext-eval: An autogenerated benchmark for evaluating large language model's ability in structure-rich text understanding. *arXiv preprint arXiv:2406.10621*.

Ahmed Hazourli. 2022. Financialbert-a pretrained language model for financial text mining. *Research Gate*, 2.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Martin Ingvar, Mathias C Blom, Casper Winsnes, Greg Robinson, Lowie Vanfleteren, and Stan Huff. 2021. On the annotation of health care pathways to allow the application of care-plans that generate data for multiple purposes. *Frontiers in Digital Health*, 3:688218.

Amirreza Jalili, Hamed Tabrizchi, Jafar Razmara, and Amir Mosavi. 2024. Bilstm for resume classification. In *2024 IEEE 22nd World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 000519–000524. IEEE.

Faizan Javed, Qinlong Luo, Matt McNair, Ferosh Jacob, Meng Zhao, and Tae Seung Kang. 2015. Carotene: A job title classification system for the online recruitment domain. In *2015 IEEE First International Conference on Big Data Computing Service and Applications*, pages 286–293. IEEE.

Kameni Florentin Flambeau Jiechieu and Norbert Tsopze. 2021. Skills prediction based on multi-label resume classification using cnn with model predictions explanation. *Neural Computing and Applications*, 33(10):5069–5087.

Guoliang Li. 2017. Human-in-the-loop data integration. *Proc. VLDB Endow.*, 10:2006–2017.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, et al. 2024. Best practices and lessons learned on synthetic data for language models. *arXiv preprint arXiv:2404.07503*.

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. *arXiv preprint arXiv:2406.15126*.

Bolanle Ojokoh and Emmanuel Adebisi. 2018. A review of question answering systems. *Journal of Web Engineering*, 17(8):717–758.

OpenAI. 2022. Chatgpt. https://openai.com/index/chatgpt/.

OpenAI. 2024. Gpt-4o. https://openai.com/index/hello-gpt-4o/.

Benjamin Paaßen. 2018. Revisiting the tree edit distance and its backtracing: A tutorial. *arXiv preprint arXiv:1805.06869*.

Dhyana Paramita. 2020. Digitalization in talent acquisition: A case study of ai in recruitment.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI Blog.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Vivek Singh Sachan, Arpana Katiyar, C Somashekher, Abhijeet Singh Chauhan, and Chaitanya Kumar Bhima. 2024. The role of artificial intelligence in hrm: Opportunities, challenges, and ethical considerations. *Educational Administration: Theory and Practice*, 30(4):7427–7435.

Timo Schick, Jane Dwivedi-Yu, Mike Lewis, Vladimir Karpukhin, Tal Schuster, Naman Goyal, Xian Chen, Wen-tau Yih Chen, Xi Victoria Gao, Wojciech Galuba, et al. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

VV Satyanarayana Tallapragada, V Sushma Raj, U Deepak, P Divya Sai, and T Mallikarjuna. 2023. Improved resume parsing based on contextual meaning extraction using bert. In *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1702–1708. IEEE.

TII Team. 2024. The falcon 3 family of open models.

Boris van Breugel and Mihaela van der Schaar. 2024. Why tabular foundation models should be a research priority. *Preprint*, arXiv:2405.01147.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint*. ArXiv:2201.11903 [cs].

Aji Prasetya Wibawa, Fachrul Kurniawan, et al. 2024. A survey of text summarization: Techniques, evaluation and challenges. *Natural Language Processing Journal*, 7:100070.

Kyra Wilson and Aylin Caliskan. 2024. Gender, race, and intersectional bias in resume screening via language model retrieval. *arXiv preprint arXiv:2407.20371*.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xinrun Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Guanglin Niu, Tongliang Li, and Zhoujun Li. 2025. Tablebench: A comprehensive and complex benchmark for table question answering. *Preprint*, arXiv:2408.09174.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2024. Large language model as attributed training data generator: A tale of diversity and bias. *Advances in Neural Information Processing Systems*, 36.

Kaizhong Zhang. 1996. A constrained edit distance between unordered labeled trees. *Algorithmica*, 15(3):205–222.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Stephen W. Huang, Jie Fu, Xiang Yue, and Wenhu Chen. 2024. Structlm: Towards building generalist models for structured knowledge grounding. *Preprint*, arXiv:2402.16671.

A Performance of LLMs on All Samples

We report the average performance across all samples in Table 4, including both successful and unsuccessful (invalid) samples. Our observations indicate that Qwen2.5 remains stronger, outperforming other models on several metrics, including the KM Ratio and ROUGE-L. However, we argue that calculating the average across all samples may not provide an accurate assessment of model performance, as it is contingent on the SR. Specifically, failed samples disproportionately lower the overall score, introducing a downward bias. This effect is particularly problematic for metrics like TED, where lower values indicate better performance, as the inclusion of failed samples can distort the results.

B Few-shot Evaluation

As summarized in Table 5 6 7 8, we have observed that providing one to three in-context examples can lead to either higher structural and semantic accuracy among successful parses or, in some settings, a reduced overall success rate. Specifically, when the prompt examples are from the same template and language domain, there is a notable improvement in semantic metrics (such as ROUGE-L and BERTScore) for successful samples, though success rates can decrease as more examples are introduced. In contrast, offering diverse examples from random templates sometimes yields a smaller gain in structural and semantic accuracy but achieves relatively higher success rates. Taken together, these findings clarify **why we have reported only zero-shot results in the main body: it presents a consistent baseline without conflating success rates with partial improvements in structural or semantic understanding.**

C How does temperature affect?

In our paper, all models—both open-source and proprietary—were evaluated under the same configuration with temperature set to 1.0. To address concerns about parameter sensitivity, we have conducted additional ablation studies[1] by varying the temperature to 0.0, 1.0, and 1.9 in Table 9 and Table 10. The results demonstrate that lower temperatures consistently yield better performance - for instance, Qwen-72B achieved a 95.6% success rate

[1]Due to computation Constraints, We randomly sample half of the Resumebench across 5 languages (250 samples per language, total 1250 samples)

Model	Type	SR ↑		KM Ratio ↑		TED ↓		ROUGE-L ↑		BERTScore ↑	
		w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON	w/ JSON	w/o JSON
Proprietary Models											
GPT-4o	API	0.9912	0.9816	0.5915	0.5900	67.91	66.16	0.6898	0.6769	0.8854	0.8792
GPT-4o-mini	API	0.9936	0.9908	0.5651	0.5591	56.35	55.91	0.6855	0.6875	0.9035	0.9015
Gemini-2.0-Flash	API	0.8840	0.8680	0.5392	0.5434	65.79	66.27	0.6072	0.6106	0.7956	0.7841
Qwen-2.5-0.5B	Local	0.9136	0.6756	0.1555	0.0915	103.50	76.18	0.3589	0.2819	0.7138	0.5359
Llama-3.2-1B	Local	0.8716	0.4640	0.3891	0.2182	78.57	41.53	0.3911	0.2087	0.6937	0.3672
Qwen-2.5-1.5B	Local	0.8044	0.6528	0.1383	0.1220	75.65	58.05	0.4614	0.3874	0.6845	0.5602
Qwen-2.5-3B	Local	0.9812	0.9160	0.5136	0.4913	63.70	57.27	0.6215	0.5928	0.8568	0.8073
Llama-3.2-3B	Local	0.7732	0.3316	0.2460	0.1554	77.51	25.26	0.4292	0.2176	0.6448	0.2908
Qwen-2.5-7B	Local	0.9888	0.9396	0.5940	0.5576	65.18	61.62	0.6865	0.6537	0.8793	0.8370
Llama3.1-8B	Local	0.9496	0.9020	0.5577	0.5360	54.34	48.23	0.6547	0.6440	0.8611	0.8273
Falcon3-7B	Local	0.948	0.9004	0.3992	0.3940	69.75	65.07	0.6496	0.6200	0.8548	0.8135
Qwen2.5-Coder-7B	Local	0.9828	0.9544	0.3469	0.3386	77.04	74.74	0.5820	0.5495	0.8430	0.8134
CodeLlama-7B	Local	0.9124	0.8992	0.3561	0.3097	82.77	83.80	0.3871	0.2878	0.6894	0.6356
Qwen-2.5-14B	Local	0.9932	0.9484	0.5784	0.5533	65.05	60.08	0.7063	0.6955	0.9163	0.8835
Qwen-2.5-32B	Local	0.9968	0.9628	0.5049	0.4810	64.56	60.63	0.6787	0.6742	0.8826	0.8780
CodeLlama-34B	Local	0.9552	0.6536	0.4964	0.3287	72.67	45.61	0.6409	0.4445	0.8574	0.5924
Qwen2.5-Coder-32B	Local	0.9972	0.9596	0.6100	0.5953	66.25	63.36	0.7010	0.6812	0.8936	0.8665
Qwen-2.5-72B	Local	0.9972	0.9612	0.6140	0.5917	61.15	55.43	0.7109	0.7066	0.9057	0.8920
Llama3.1-70B	Local	0.9708	0.9580	0.5468	0.5284	61.97	60.73	0.7070	0.7020	0.8938	0.8882
Ministral-8B-2410	Local	0.9004	0.8292	0.2750	0.2578	67.76	61.59	0.5595	0.5132	0.7923	0.7273
Mistral-Small-24B-2501	Local	0.9924	0.9580	0.5842	0.5672	64.81	61.60	0.6905	0.6776	0.8830	0.8638
Reasoning Models											
DeepSeek-R1-Distill-Qwen-1.5B	Local	0.7236	0.5824	0.3030	0.2447	76.89	55.96	0.2737	0.2315	0.5523	0.4540
DeepSeek-R1-Distill-Qwen-7B	Local	0.8964	0.8096	0.4418	0.4040	65.87	61.24	0.5734	0.5055	0.7899	0.7087
DeepSeek-R1-Distill-Llama-8B	Local	0.7800	0.9448	0.4168	0.5333	51.23	59.35	0.5199	0.6377	0.6956	0.8498

Table 4: Performance of LLMs on ResumeBench. **The Metrics Are Computed as The Average Score across All Samples.**

Model Name	Type	Success Rate ↑
Qwen-2.5-7B (One-Shot)	Local	0.3683
Qwen-2.5-7B (Two-Shot)	Local	0.3889
Qwen-2.5-7B (Three-Shot)	Local	0.7964
Qwen-2.5-14B (One-Shot)	Local	0.5967
Qwen-2.5-14B (Two-Shot)	Local	0.4802
Qwen-2.5-14B (Three-Shot)	Local	0.9944
Qwen-2.5-32B (One-Shot)	Local	0.8122
Qwen-2.5-32B (Two-Shot)	Local	0.7882
Qwen-2.5-32B (Three-Shot)	Local	0.8449
Qwen-2.5-72B (One-Shot)	Local	0.8445
Qwen-2.5-72B (Two-Shot)	Local	0.8283
Qwen-2.5-72B (Three-Shot)	Local	0.8081

Table 5: Average on all samples (success + failed), example samples from same template but different domains

Model Name	Type	Success Rate ↑
Qwen-2.5-7B (One-Shot)	Local	0.5445
Qwen-2.5-7B (Two-Shot)	Local	0.4656
Qwen-2.5-7B (Three-Shot)	Local	0.2960
Qwen-2.5-14B (One-Shot)	Local	0.9443
Qwen-2.5-14B (Two-Shot)	Local	0.9283
Qwen-2.5-14B (Three-Shot)	Local	0.4562
Qwen-2.5-32B (One-Shot)	Local	0.8423
Qwen-2.5-32B (Two-Shot)	Local	0.8480
Qwen-2.5-32B (Three-Shot)	Local	0.7523
Qwen-2.5-72B (One-Shot)	Local	0.9762
Qwen-2.5-72B (Two-Shot)	Local	0.9443
Qwen-2.5-72B (Three-Shot)	Local	0.9364

Table 6: Average on all samples (success + failed), example samples randomly

at temperature 0.0 versus 94.1% at 1.9, with similar trends observed across all metrics (KM Ratio, TED, ROUGE-L, and BERTScore). Nevertheless, certain models still demonstrate robust performance even at higher temperatures, indicating that temperature alone does not fully resolve ambiguity or structural complexity in resume parsing.

D Vision Language Models

In the following additional experiments[1][2] (Table 11 and Table 12), we feed each resume as an image (PNG) plus a text-based prompt to vision-language models (VLMs). Our results indicate that vision-language models achieve higher parsing accuracy and, for successful parses, noticeably better performance in both structural (KM Ratio, TED) and semantic (ROUGE-L, BERTScore) metrics compared to their text-only counterparts.

This finding underscores that document layout is not merely a cosmetic element. Even though we rely on text extraction in our main evaluation, the arrangement of sections, headers, and nested blocks—which originates from the template designs—still affects how the extracted text is grouped and interpreted. VLMs that directly process the visual layout appear to benefit from these visual cues, demonstrating a more robust grasp of both structure and meaning.

However, multimodal setups introduce additional complexities (e.g., image resolution), making them a larger undertaking than text-based experiments alone. Our main paper focuses on text-based parsing to establish a consistent, reproducible baseline, but we plan to delve deeper into multi-modal

Model Name	Type	KM Ratio ↑	TED ↓	ROUGE-L ↑	BERTScore ↑
Qwen-2.5-7B (One-Shot)	Local	0.7297	52.9130	0.7610	0.9050
Qwen-2.5-7B (Two-Shot)	Local	0.6987	57.2990	0.7074	0.8971
Qwen-2.5-7B (Three-Shot)	Local	0.7332	48.7789	0.7558	0.9053
Qwen-2.5-14B (One-Shot)	Local	0.7861	37.1678	0.7770	0.9367
Qwen-2.5-14B (Two-Shot)	Local	0.7722	39.8500	0.7464	0.9303
Qwen-2.5-14B (Three-Shot)	Local	0.7852	38.1992	0.8025	0.9439
Qwen-2.5-32B (One-Shot)	Local	0.8784	20.6010	0.8020	0.9427
Qwen-2.5-32B (Two-Shot)	Local	0.9074	12.3959	0.8532	0.9530
Qwen-2.5-32B (Three-Shot)	Local	0.9246	9.5782	0.8489	0.9607
Qwen-2.5-72B (One-Shot)	Local	0.9048	24.2938	0.8079	0.9497
Qwen-2.5-72B (Two-Shot)	Local	0.9063	17.8357	0.8500	0.9629
Qwen-2.5-72B (Three-Shot)	Local	0.9241	11.8812	0.8872	0.9680

Table 7: Average on all samples (success + failed), example samples from same template but different domains

Model Name	Type	Success Rate ↑	KM Ratio ↑	TED ↓	ROUGE-L ↑	BERTScore ↑
Qwen-2.5-7B (One-Shot)	Local	0.5445	0.6515	71.6148	0.6965	0.9174
Qwen-2.5-7B (Two-Shot)	Local	0.4656	0.5720	68.0000	0.7213	0.9288
Qwen-2.5-7B (Three-Shot)	Local	0.2960	0.5740	78.2973	0.6829	0.8898
Qwen-2.5-14B (One-Shot)	Local	0.9443	0.6636	52.4746	0.7371	0.9301
Qwen-2.5-14B (Two-Shot)	Local	0.9283	0.6754	53.5000	0.7408	0.9319
Qwen-2.5-14B (Three-Shot)	Local	0.4562	0.6974	53.1053	0.7507	0.9360
Qwen-2.5-32B (One-Shot)	Local	0.8423	0.7008	42.2002	0.7441	0.9276
Qwen-2.5-32B (Two-Shot)	Local	0.8480	0.7179	42.2973	0.7526	0.9336
Qwen-2.5-32B (Three-Shot)	Local	0.7523	0.7123	43.8191	0.7612	0.9354
Qwen-2.5-72B (One-Shot)	Local	0.9762	0.7267	38.0821	0.7576	0.9393
Qwen-2.5-72B (Two-Shot)	Local	0.9443	0.7393	36.0254	0.7773	0.9432
Qwen-2.5-72B (Three-Shot)	Local	0.9364	0.7389	37.2906	0.7780	0.9433

Table 8: Average on Success Samples, example samples randomly

Model Name	Type	Success Rate ↑
Qwen-2.5-7B (0.0)	Local	0.924
Qwen-2.5-7B (1.0)	Local	0.920
Qwen-2.5-7B (1.9)	Local	0.864
Qwen-2.5-14B (0.0)	Local	0.944
Qwen-2.5-14B (1.0)	Local	0.944
Qwen-2.5-14B (1.9)	Local	0.904
Qwen-2.5-32B (0.0)	Local	0.956
Qwen-2.5-32B (1.0)	Local	0.960
Qwen-2.5-32B (1.9)	Local	0.924
Qwen-2.5-72B (0.0)	Local	0.956
Qwen-2.5-72B (1.0)	Local	0.952
Qwen-2.5-72B (1.9)	Local	0.941

Table 9: Average on all samples (success + failed)

analysis in future work. We believe these initial multimodal results already illustrate why investing in template design and layout is valuable: even when text-based extraction is available, the visual format of a resume offers meaningful signals that large language models can exploit for more accurate parsing.

[2]We choose Qwen-2.5-VL Series since they provide various model size options and they are built on corresponding Qwen-2.5 LLM (evaluated in our paper)

E Qualitative Error Analysis

E.1 Language Scope

As detailed in Section 3.1 and Section 3.2.1, while human review is part of template design, the pipeline's modular architecture (templates + automated content generation) is inherently extensible to new languages once templates are validated.

Our focus on English, Chinese, Spanish, French, and German reflects their prevalence in global hiring markets and the availability of standardized resume formats. Low-resource language challenges (e.g., template scarcity, cultural validation) are inherent to template design (Section 3.1), not pipeline limitations. These challenges necessitate rigorous human review for template and content alignment, which falls beyond the scope of this initial benchmark.

Critically, our pipeline's scalability ensures that once validated templates and terminology are established for a new language, synthetic resumes can be generated at scale without additional human effort.

Model Name	Type	KM Ratio ↑	TED ↓	ROUGE-L ↑	BERTScore ↑
Qwen-2.5-7B (0.0)	Local	0.5750	71.1212	0.6660	0.8829
Qwen-2.5-7B (1.0)	Local	0.5758	71.3652	0.6756	0.8805
Qwen-2.5-7B (1.9)	Local	0.5669	74.0185	0.6501	0.8698
Qwen-2.5-14B (0.0)	Local	0.5864	63.6780	0.7407	0.9338
Qwen-2.5-14B (1.0)	Local	0.5814	64.7500	0.7370	0.9309
Qwen-2.5-14B (1.9)	Local	0.5771	64.8186	0.7338	0.9292
Qwen-2.5-32B (0.0)	Local	0.5015	62.4854	0.7147	0.9176
Qwen-2.5-32B (1.0)	Local	0.5167	62.1833	0.7102	0.9119
Qwen-2.5-32B (1.9)	Local	0.4968	67.2251	0.6651	0.8959
Qwen-2.5-72B (0.0)	Local	0.6135	57.1506	0.7450	0.9294
Qwen-2.5-72B (1.0)	Local	0.6058	58.6261	0.7398	0.9268
Qwen-2.5-72B (1.9)	Local	0.5922	59.0596	0.7378	0.9294

Table 10: Average on Success Samples

Model Name	Type	Success Rate ↑
Qwen-2.5-VL-3B	Local	0.9764
Qwen-2.5-VL-7B	Local	0.9848
Qwen-2.5-VL-32B	Local	0.9762
Qwen-2.5-VL-72B	Local	0.9920

Table 11: Average on all samples (success + failed)

E.2 Bias Analysis

Unlike hiring algorithms, parsing extracts neutral fields (e.g., 'Degree', 'Company') without subjective interpretation. Our synthetic data avoids embedding biases by design, and occupational diversity is ensured via 30 career domains (please refer to the description in Figure 1).

E.3 Failure Analysis

We categorized and described model failures, identifying two distinct types:

- The first is complete parsing failure, where models either produce invalid JSON (e.g., missing brackets or commas) or outright refuse to parse, sometimes indicating the need for more text despite having received the full prompt. This issue is discussed explicitly in Section 4.3, addressing the success rate metric used to quantify the conversion success of outputs into valid JSON formats.
- The second concerns partially successful parses that include structural or semantic inaccuracies (detailed in Section 4.3 and Section 4.4). Structurally, resumes can lose their hierarchical alignment or nested structure when, for instance, having several roles under "Work Experience," each containing job titles, dates, and multiple bullet points or nested descriptions. Models often confuse or omit elements in these parallel blocks, reducing scores on our structural metrics. Semantically, certain bullet points may be truncated or rephrased, leading to incomplete or misrepresented content.
- Nested sections and complex domain-specific details (e.g., finance, education, public administration) also tend to be challenging for smaller models, which show more frequent incomplete parses in business or social science domains compared to science or engineering fields.

E.4 Time cost and budget on the template creation, annotation and validation process

- During the template creation stage, each template required an average of about 5–10 minutes of manual adjustment to ensure proper HTML/CSS formatting.
- For annotation and data generation, we used GPT-4o with a total expenditure of around $300 (per dataset), covering prompts for multiple languages and career domains.
- Lastly, we verified that each generated resume matched the intended structure; The pass rate of the generated samples is 96%, with most samples requiring no human augmentation.

F Success Rate (SR) Supplementary Materials: language-wise model performance on Success Rate (SR)

Success Rate (SR) supplementary materials we would like to supplement language-wise model per-

Model Name	Type	KM Ratio ↑	TED ↓	ROUGE-L ↑	BERTScore ↑
Qwen-2.5-VL-3B	Local	0.5843	66.7681	0.7122	0.9214
Qwen-2.5-VL-7B	Local	0.5439	64.6296	0.7187	0.9247
Qwen-2.5-VL-32B	Local	0.5757	66.4467	0.6874	0.9127
Qwen-2.5-VL-72B	Local	0.6335	60.7903	0.7529	0.9410

Table 12: Average on Success Samples

formance on Success Rate (SR) for incomplete failure case analysis, see Figure 13.

G KM Ratio Supplementary Materials: language-wise model performance on KM Radio.

KM Ratio supplementary materials we would like to supplement language-wise model performance on KM Radio for incomplete failure case analysis, see Figure 14. The percentage indicates the success rate for each individual language.

H Tree Edit Distance Supplementary Materials: language-wise model performance on tree edit distance

Tree Edit Distance supplementary materials we would like to supplement language-wise model performance on tree edit distance for incomplete failure case analysis, see Figure 15. The percentage indicates the success rate for each individual language.

I ROUGE-L Supplementary Materials: language-wise model performance on ROUGE-L

ROUGE-L supplementary materials we would like to supplement language-wise model performance on ROUGE-L for incomplete failure case analysis, see Figure 16. The percentage indicates the success rate for each individual language.

J Template-wise Supplementary Materials

We found that the performance of most templates fluctuates around the average, see Figure 17. Among them, Template 7 underperforms significantly (-12.23% below the average). It is a single-column layout with numerous sections and a tendency to include deeply nested experience entries. Template 48 also shows weaker performance (-11.15% below the average). It features a two-column layout with an approximate 3:7 ratio between the left and right columns. The left column contains dense personal information and language details, while the right focuses on experience. Overall, its structure is more complex than standard single- or two-column templates.

K BERTScore Supplementary Materials: language-wise model performance on BERTScore

BERTScore Supplementary Materials we would like to supplement language-wise model performance on BERTScore, see Figure 18. The percentage indicates the success rate for each individual language

L Computational Efficiency Metrics

While our primary focus is to evaluate structural and semantic accuracy across multilingual resumes, a challenge underexplored in prior work, we found that efficiency is critical for real-world deployment in time-sensitive recruitment systems. To address this, we added a new parsing-time table, see Figure 19, reporting average inference latency across all models in our benchmark.

All open-source models were evaluated on 4× NVIDIA H100 (80GB) GPUs in a consistent setup. Our observations confirm that larger models and reasoning-focused models generally incur higher latency, with Qwen-2.5-72B and DeepSeek-R1-Distill showing longer runtimes. Enabling JSON-mode will add additional latency.

Please note: We took Huggingface Transformers to run all experiments (without any inference acceleration); Using frameworks like vLLM can accelerate the inference process. We use a batch size equals to 10 to run all experiments, therefore there are higher latency than processing one single sample

Speed-Performance Trade-off Suggestion: we find that Qwen-2.5-14B offers a strong trade-off, achieving near-72B performance on most accuracy metrics while significantly reducing computational cost. We will add a note in Section 5.1 to recommend Qwen-2.5-14B as a practical baseline for

Model	German	English	Spanish	French	Chinese
CodeLlama-34b-Instruct-hf	0.526	0.808	0.722	0.628	0.584
CodeLlama-34b-Instruct-hf_json_mode	0.954	0.980	0.956	0.954	0.932
CodeLlama-7b-Instruct-hf	0.920	0.894	0.900	0.934	0.848
CodeLlama-7b-Instruct-hf_json_mode	0.920	0.866	0.876	0.918	0.982
DeepSeek-R1-Distill-Llama-8B	0.930	0.940	0.964	0.952	0.938
DeepSeek-R1-Distill-Llama-8B_json_mode	0.762	0.816	0.784	0.762	0.776
DeepSeek-R1-Distill-Qwen-1.5B	0.602	0.622	0.584	0.638	0.466
DeepSeek-R1-Distill-Qwen-1.5B_json_mode	0.664	0.790	0.682	0.752	0.730
DeepSeek-R1-Distill-Qwen-7B	0.854	0.834	0.792	0.804	0.764
DeepSeek-R1-Distill-Qwen-7B_json_mode	0.890	0.910	0.870	0.892	0.920
Falcon3-7B-Instruct	0.890	0.966	0.894	0.868	0.884
Falcon3-7B-Instruct_json_mode	0.942	0.990	0.942	0.906	0.960
GPT-4o	0.984	0.962	0.982	0.984	0.996
GPT-4o-mini	0.998	0.996	0.994	0.984	0.982
GPT-4o-mini_json_mode	0.996	0.984	0.998	0.994	0.996
GPT-4o_json_mode	0.984	0.982	0.998	0.998	0.994
Gemini-2.0-Flash	0.920	0.900	0.740	0.880	0.900
Gemini-2.0-Flash_json_mode	0.920	0.920	0.780	0.900	0.900
Llama-3.1-70B-Instruct	0.940	0.954	0.950	0.964	0.982
Llama-3.1-70B-Instruct_json_mode	0.966	0.968	0.958	0.968	0.994
Llama-3.1-8B-Instruct	0.894	0.928	0.892	0.912	0.884
Llama-3.1-8B-Instruct_json_mode	0.958	0.972	0.948	0.970	0.900
Llama-3.2-1B-Instruct	0.470	0.450	0.408	0.388	0.604
Llama-3.2-1B-Instruct_json_mode	0.880	0.934	0.800	0.810	0.934
Llama-3.2-3B-Instruct	0.280	0.348	0.224	0.254	0.552
Llama-3.2-3B-Instruct_json_mode	0.752	0.816	0.718	0.758	0.822
Ministral-8B-Instruct-2410	0.874	0.824	0.802	0.796	0.850
Ministral-8B-Instruct-2410_json_mode	0.922	0.922	0.870	0.878	0.910
Mistral-Small-24B-Instruct-2501	0.952	0.968	0.948	0.976	0.946
Mistral-Small-24B-Instruct-2501_json_mode	0.992	0.994	0.998	1.000	0.978
Qwen2.5-0.5B-Instruct	0.646	0.692	0.664	0.636	0.740
Qwen2.5-0.5B-Instruct_json_mode	0.912	0.920	0.908	0.900	0.928
Qwen2.5-1.5B-Instruct	0.670	0.700	0.606	0.604	0.684
Qwen2.5-1.5B-Instruct_json_mode	0.814	0.852	0.760	0.750	0.846
Qwen2.5-14B-Instruct	0.928	0.940	0.934	0.952	0.988
Qwen2.5-14B-Instruct_json_mode	0.992	0.994	0.992	0.996	0.992
Qwen2.5-32B-Instruct	0.946	0.958	0.956	0.960	0.994
Qwen2.5-32B-Instruct_json_mode	0.994	0.998	0.998	1.000	0.994
Qwen2.5-3B-Instruct	0.930	0.924	0.874	0.902	0.950
Qwen2.5-3B-Instruct_json_mode	0.986	0.982	0.972	0.976	0.990
Qwen2.5-72B-Instruct	0.950	0.950	0.952	0.960	0.994
Qwen2.5-72B-Instruct_json_mode	0.994	0.996	0.996	1.000	1.000
Qwen2.5-7B-Instruct	0.942	0.960	0.950	0.928	0.918
Qwen2.5-7B-Instruct_json_mode	0.986	0.986	0.986	0.990	0.996
Qwen2.5-Coder-32B-Instruct	0.962	0.956	0.960	0.958	0.962
Qwen2.5-Coder-32B-Instruct_json_mode	0.998	1.000	0.996	1.000	0.992
Qwen2.5-Coder-7B-Instruct	0.950	0.976	0.954	0.956	0.936
Qwen2.5-Coder-7B-Instruct_json_mode	0.994	0.994	0.970	0.988	0.968

Table 13: Success Rate (SR) Supplementary Materials: language-wise model performance on Success Rate (SR)

applications that balance accuracy and latency.

M Parsing Schema

We define two distinct parsing schemas for Resumebench and Resumebench-Mix, respectively. For Resumebench, we employ a comprehensive schema designed to exhaustively extract as much information as possible from the resumes. In contrast, for Resumebench-Mix, we adopt a more streamlined schema that focuses on parsing only the most essential information—namely, contact details, education, and work experience—due to the high cost

Model	German	English	Spanish	French	Chinese
CodeLlama-34b-Instruct-hf	0.5113	0.5348	0.5382	0.5014	0.4095
CodeLlama-34b-Instruct-hf_json_mode	0.5676	0.5730	0.5093	0.5005	0.4449
CodeLlama-7b-Instruct-hf	0.3208	0.3155	0.3271	0.3316	0.4329
CodeLlama-7b-Instruct-hf_json_mode	0.3626	0.3215	0.3800	0.3785	0.4970
DeepSeek-R1-Distill-Llama-8B	0.5522	0.5881	0.5689	0.5665	0.5464
DeepSeek-R1-Distill-Llama-8B_json_mode	0.4856	0.5487	0.5548	0.5574	0.5239
DeepSeek-R1-Distill-Qwen-1.5B	0.4557	0.4425	0.4195	0.4281	0.3346
DeepSeek-R1-Distill-Qwen-1.5B_json_mode	0.4271	0.4520	0.3833	0.3948	0.4331
DeepSeek-R1-Distill-Qwen-7B	0.4912	0.4936	0.4944	0.5176	0.4985
DeepSeek-R1-Distill-Qwen-7B_json_mode	0.4980	0.5027	0.4817	0.4844	0.4969
Falcon3-7B-Instruct	0.4402	0.4828	0.3106	0.4718	0.4806
Falcon3-7B-Instruct_json_mode	0.4186	0.4741	0.3240	0.4560	0.4312
Llama-3.1-70B-Instruct	0.5245	0.5490	0.5515	0.5536	0.5778
Llama-3.1-70B-Instruct_json_mode	0.5437	0.5652	0.5638	0.5586	0.5845
Llama-3.1-8B-Instruct	0.6073	0.6283	0.5808	0.5791	0.5746
Llama-3.1-8B-Instruct_json_mode	0.5982	0.6196	0.5707	0.5815	0.5643
Llama-3.2-1B-Instruct	0.4887	0.5128	0.3950	0.4326	0.4994
Llama-3.2-1B-Instruct_json_mode	0.4522	0.4981	0.3524	0.4150	0.4971
Llama-3.2-3B-Instruct	0.3973	0.4660	0.4512	0.4768	0.5100
Llama-3.2-3B-Instruct_json_mode	0.2670	0.3338	0.2737	0.2971	0.4077
Ministral-8B-Instruct-2410	0.3127	0.3099	0.2981	0.2953	0.3369
Ministral-8B-Instruct-2410_json_mode	0.3067	0.3041	0.2952	0.2902	0.3299
Mistral-Small-24B-Instruct-2501	0.6228	0.5951	0.5636	0.5917	0.5867
Mistral-Small-24B-Instruct-2501_json_mode	0.6124	0.5875	0.5639	0.5917	0.5879
Qwen2.5-0.5B-Instruct	0.1514	0.1009	0.1737	0.1785	0.0828
Qwen2.5-0.5B-Instruct_json_mode	0.1992	0.1532	0.1920	0.2081	0.1005
Qwen2.5-1.5B-Instruct	0.2075	0.2148	0.1615	0.1955	0.1532
Qwen2.5-1.5B-Instruct_json_mode	0.1841	0.1820	0.1609	0.1814	0.1513
Qwen2.5-14B-Instruct	0.5812	0.6047	0.5596	0.5752	0.5959
Qwen2.5-14B-Instruct_json_mode	0.5768	0.6056	0.5604	0.5686	0.6002
Qwen2.5-32B-Instruct	0.5342	0.5563	0.4768	0.4730	0.4598
Qwen2.5-32B-Instruct_json_mode	0.5413	0.5644	0.4930	0.4794	0.4544
Qwen2.5-3B-Instruct	0.5470	0.5528	0.5276	0.5458	0.5090
Qwen2.5-3B-Instruct_json_mode	0.5414	0.5430	0.5113	0.5158	0.5059
Qwen2.5-72B-Instruct	0.6235	0.6349	0.6047	0.6141	0.6014
Qwen2.5-72B-Instruct_json_mode	0.6199	0.6315	0.6078	0.6165	0.6032
Qwen2.5-7B-Instruct	0.5981	0.6059	0.5921	0.5712	0.5995
Qwen2.5-7B-Instruct_json_mode	0.6071	0.6092	0.6040	0.5724	0.6109
Qwen2.5-Coder-32B-Instruct	0.6172	0.6369	0.6155	0.6116	0.6205
Qwen2.5-Coder-32B-Instruct_json_mode	0.6126	0.6255	0.6059	0.6048	0.6098
Qwen2.5-Coder-7B-Instruct	0.3623	0.3615	0.3510	0.3440	0.3548
Qwen2.5-Coder-7B-Instruct_json_mode	0.3604	0.3649	0.3466	0.3365	0.3564
gemini-2.0-flash	0.6260	0.6219	0.5879	0.6292	0.6585
gemini-2.0-flash_json_mode	0.6238	0.6186	0.5735	0.6090	0.6197
gpt-4o-mini	0.6185	0.5837	0.5877	0.5452	0.4808
gpt-4o-mini_json_mode	0.6098	0.5955	0.5751	0.5639	0.4932
gpt-4o	0.6250	0.6155	0.5955	0.5802	0.5942
gpt-4o_json_mode	0.6006	0.6043	0.5841	0.5909	0.6017

Table 14: KM Ratio Supplementary Materials: language-wise model performance on KM Radio

of annotation. This difference in schema design helps explain why the performance of the LLM on synthetic data in Resumebench-Mix exceeds its performance in Resumebench. The introduction of a more common, simplified parsing schema reduces the density of the extracted information, thereby making the task less complex for the model.

Example Parsing Schema for Resumebench

Listing 1: Example Parsing Schema for Resumebench

```
{
    "{{NAME}}": "NAME",
```

Model	German	English	Spanish	French	Chinese
CodeLlama-34b-Instruct-hf	69.3	61.3	71.4	67.9	82.0
CodeLlama-34b-Instruct-hf_json_mode	71.0	62.4	78.5	75.5	93.8
CodeLlama-7b-Instruct-hf	95.6	86.1	98.9	96.1	88.9
CodeLlama-7b-Instruct-hf_json_mode	89.0	84.5	98.7	95.9	85.8
DeepSeek-R1-Distill-Llama-8B	64.2	53.4	64.1	63.4	69.0
DeepSeek-R1-Distill-Llama-8B_json_mode	70.0	57.6	66.4	65.2	69.8
DeepSeek-R1-Distill-Qwen-1.5B	101.0	82.7	103.7	99.7	93.1
DeepSeek-R1-Distill-Qwen-1.5B_json_mode	111.9	94.4	111.6	110.7	104.5
DeepSeek-R1-Distill-Qwen-7B	79.7	63.1	80.3	79.2	76.3
DeepSeek-R1-Distill-Qwen-7B_json_mode	77.0	62.2	78.7	79.1	71.0
Falcon3-7B-Instruct	72.8	60.3	76.8	69.1	83.4
Falcon3-7B-Instruct_json_mode	75.2	61.7	77.7	70.4	83.2
Llama-3.1-70B-Instruct	62.3	56.2	64.4	61.2	72.6
Llama-3.1-70B-Instruct_json_mode	62.6	57.5	64.8	61.7	72.3
Llama-3.1-8B-Instruct	50.9	44.7	54.6	52.8	64.8
Llama-3.1-8B-Instruct_json_mode	55.1	47.5	60.1	57.6	66.5
Llama-3.2-1B-Instruct	91.4	73.4	102.1	91.5	90.3
Llama-3.2-1B-Instruct_json_mode	92.6	74.3	102.3	93.8	90.1
Llama-3.2-3B-Instruct	82.7	71.7	80.0	74.0	75.1
Llama-3.2-3B-Instruct_json_mode	105.9	92.5	111.3	105.5	88.3
Ministral-8B-Instruct-2410	75.0	68.9	77.4	78.1	72.3
Ministral-8B-Instruct-2410_json_mode	76.2	69.0	79.9	78.5	73.1
Mistral-Small-24B-Instruct-2501	61.4	57.1	65.1	61.2	77.0
Mistral-Small-24B-Instruct-2501_json_mode	62.3	58.5	66.9	61.5	77.5
Qwen2.5-0.5B-Instruct	115.6	102.1	117.0	115.4	114.2
Qwen2.5-0.5B-Instruct_json_mode	115.5	102.0	118.7	116.2	114.3
Qwen2.5-1.5B-Instruct	90.9	76.8	97.9	89.0	91.3
Qwen2.5-1.5B-Instruct_json_mode	96.0	81.1	103.0	93.8	97.4
Qwen2.5-14B-Instruct	62.6	56.4	65.7	63.0	68.8
Qwen2.5-14B-Instruct_json_mode	65.8	58.8	68.3	65.3	69.3
Qwen2.5-32B-Instruct	61.4	54.6	65.0	62.5	71.0
Qwen2.5-32B-Instruct_json_mode	63.9	56.2	67.8	64.7	71.4
Qwen2.5-3B-Instruct	62.9	55.6	66.2	62.2	65.7
Qwen2.5-3B-Instruct_json_mode	64.9	57.5	69.1	66.3	66.9
Qwen2.5-72B-Instruct	54.5	50.5	58.5	55.3	69.1
Qwen2.5-72B-Instruct_json_mode	58.5	55.0	62.6	59.9	70.5
Qwen2.5-7B-Instruct	65.5	58.7	67.4	64.6	71.9
Qwen2.5-7B-Instruct_json_mode	66.1	58.9	67.1	65.0	72.4
Qwen2.5-Coder-32B-Instruct	65.4	59.6	67.6	64.0	73.5
Qwen2.5-Coder-32B-Instruct_json_mode	65.8	60.4	68.3	64.7	73.0
Qwen2.5-Coder-7B-Instruct	79.1	71.6	85.6	80.4	75.1
Qwen2.5-Coder-7B-Instruct_json_mode	79.6	71.1	85.2	80.8	75.4
gemini-2.0-flash	73.2	67.2	79.7	77.9	84.5
gemini-2.0-flash_json_mode	67.8	64.2	80.3	77.1	83.9
gpt-4o-mini	52.7	51.5	55.9	54.8	67.2
gpt-4o-mini_json_mode	54.9	50.3	55.2	53.0	69.3
gpt-4o	63.2	60.4	67.7	63.7	82.2
gpt-4o_json_mode	64.6	62.0	70.0	64.5	81.0

Table 15: Tree Edit Distance Supplementary Materials: language-wise model performance on tree edit distance

```
"field-contact": [
    {
        "{{LOCATION}}": "LOCATION",
        "{{PHONE}}": "PHONE",
        "{{EMAIL}}": "EMAIL",
        "{{LINKEDIN}}": "LINKEDIN"
    }
],
"{{SUMMARY}}": "SUMMARY",
"Work Experience": [
    {
        "{{START_DATE}}": "
            START_DATE",
        "{{END_DATE}}": "END_DATE",
        "{{JOB_TITLE}}": "JOB_TITLE"
            ,
```

Model	German	English	Spanish	French	Chinese
CodeLlama-34b-Instruct-hf	0.6689	0.6799	0.6535	0.6558	0.7495
CodeLlama-34b-Instruct-hf_json_mode	0.6783	0.6686	0.6498	0.6531	0.7061
CodeLlama-7b-Instruct-hf	0.3000	0.3151	0.2869	0.2757	0.4314
CodeLlama-7b-Instruct-hf_json_mode	0.4500	0.4515	0.3660	0.3235	0.5222
DeepSeek-R1-Distill-Llama-8B	0.6490	0.6957	0.6508	0.6534	0.7265
DeepSeek-R1-Distill-Llama-8B_json_mode	0.6356	0.6794	0.6595	0.6347	0.7218
DeepSeek-R1-Distill-Qwen-1.5B	0.2773	0.5250	0.3102	0.2828	0.6485
DeepSeek-R1-Distill-Qwen-1.5B_json_mode	0.2371	0.4968	0.2902	0.2521	0.5908
DeepSeek-R1-Distill-Qwen-7B	0.5696	0.6691	0.5903	0.5874	0.7108
DeepSeek-R1-Distill-Qwen-7B_json_mode	0.5961	0.6580	0.6124	0.6177	0.7108
Falcon3-7B-Instruct	0.6736	0.6898	0.6772	0.6804	0.7217
Falcon3-7B-Instruct_json_mode	0.6710	0.6889	0.6794	0.6764	0.7094
Llama-3.1-70B-Instruct	0.7387	0.7318	0.7291	0.7188	0.7454
Llama-3.1-70B-Instruct_json_mode	0.7317	0.7238	0.7215	0.7150	0.7488
Llama-3.1-8B-Instruct	0.7212	0.7172	0.7037	0.7066	0.7213
Llama-3.1-8B-Instruct_json_mode	0.6937	0.6924	0.6701	0.6714	0.7217
Llama-3.2-1B-Instruct	0.4148	0.5003	0.3825	0.3653	0.5393
Llama-3.2-1B-Instruct_json_mode	0.4196	0.5046	0.3733	0.3780	0.5462
Llama-3.2-3B-Instruct	0.6250	0.6678	0.6367	0.6153	0.6914
Llama-3.2-3B-Instruct_json_mode	0.5329	0.5615	0.5233	0.4952	0.6520
Ministral-8B-Instruct-2410	0.6348	0.6127	0.6046	0.6025	0.6376
Ministral-8B-Instruct-2410_json_mode	0.6405	0.6156	0.6076	0.6126	0.6295
Mistral-Small-24B-Instruct-2501	0.7147	0.7021	0.7116	0.6910	0.7175
Mistral-Small-24B-Instruct-2501_json_mode	0.7023	0.6909	0.6925	0.6864	0.7070
Qwen2.5-0.5B-Instruct	0.3328	0.4980	0.3372	0.3082	0.5812
Qwen2.5-0.5B-Instruct_json_mode	0.3154	0.4795	0.3159	0.2971	0.5509
Qwen2.5-1.5B-Instruct	0.5446	0.6103	0.5691	0.5667	0.6696
Qwen2.5-1.5B-Instruct_json_mode	0.5256	0.5985	0.5436	0.5502	0.6424
Qwen2.5-14B-Instruct	0.7406	0.7304	0.7319	0.7259	0.7381
Qwen2.5-14B-Instruct_json_mode	0.7110	0.7073	0.7014	0.6954	0.7405
Qwen2.5-32B-Instruct	0.6970	0.7085	0.7149	0.6982	0.6834
Qwen2.5-32B-Instruct_json_mode	0.6759	0.6879	0.6880	0.6793	0.6734
Qwen2.5-3B-Instruct	0.6447	0.6520	0.6272	0.6232	0.6859
Qwen2.5-3B-Instruct_json_mode	0.6323	0.6402	0.6045	0.6035	0.6859
Qwen2.5-72B-Instruct	0.7453	0.7372	0.7212	0.7260	0.7455
Qwen2.5-72B-Instruct_json_mode	0.7163	0.7077	0.6927	0.7011	0.7464
Qwen2.5-7B-Instruct	0.6951	0.6953	0.6829	0.6744	0.7315
Qwen2.5-7B-Instruct_json_mode	0.6864	0.6956	0.6858	0.6704	0.7327
Qwen2.5-Coder-32B-Instruct	0.7050	0.7069	0.6968	0.6980	0.7425
Qwen2.5-Coder-32B-Instruct_json_mode	0.6982	0.6946	0.6927	0.6888	0.7409
Qwen2.5-Coder-7B-Instruct	0.5877	0.5733	0.5329	0.5517	0.6345
Qwen2.5-Coder-7B-Instruct_json_mode	0.6013	0.5916	0.5526	0.5710	0.6447
gemini-2.0-flash	0.7076	0.7034	0.6749	0.6956	0.7303
gemini-2.0-flash_json_mode	0.7137	0.6958	0.6519	0.6652	0.7026
gpt-4o-mini	0.7148	0.6859	0.6950	0.6713	0.6976
gpt-4o-mini_json_mode	0.6972	0.6970	0.7002	0.6744	0.6727
gpt-4o	0.7095	0.7157	0.6810	0.6738	0.6746
gpt-4o_json_mode	0.7077	0.7022	0.6775	0.6692	0.7204

Table 16: ROUGE-L Supplementary Materials: language-wise model performance on ROUGE-L


```
"{{COMPANY}}": "COMPANY",
"{{LOCATION}}": "LOCATION",
"{{WORK_DESCRIPTION-STR-
    MULTIPLE-3}}": [
    {
        "WORK_DESCRIPTION-
            STR-0": "
            WORK_DESCRIPTION
            -0"
    },
    {
        "WORK_DESCRIPTION-
            STR-1": "
            WORK_DESCRIPTION
            -1"
    },
```

Template	Success Rate (%)	Deviation from Average
Template 1	93.99	+5.99%
Template 2	79.94	-8.06%
Template 3	88.08	+0.08%
Template 4	85.45	-2.55%
Template 5	89.30	+1.30%
Template 6	87.12	-0.88%
Template 7	75.77	-12.23%
Template 8	88.92	+0.92%
Template 9	88.62	+0.62%
Template 10	87.87	-0.13%
Template 11	87.84	-0.16%
Template 12	87.37	-0.63%
Template 13	86.08	-1.92%
Template 14	86.98	-1.02%
Template 15	87.89	-0.11%
Template 16	90.42	+2.42%
Template 17	92.52	+4.52%
Template 18	87.56	-0.44%
Template 19	89.01	+1.01%
Template 20	92.52	+4.52%
Template 21	87.81	-0.19%
Template 22	88.26	+0.26%
Template 23	93.97	+5.97%
Template 24	89.25	+1.25%
Template 25	86.38	-1.62%
Template 26	89.28	+1.28%
Template 27	89.96	+1.96%
Template 28	88.37	+0.37%
Template 29	90.42	+2.42%
Template 30	94.63	+6.63%
Template 31	94.80	+6.80%
Template 32	94.11	+6.11%
Template 33	87.56	-0.44%
Template 34	88.12	+0.12%
Template 35	86.67	-1.33%
Template 36	85.35	-2.65%
Template 37	89.98	+1.98%
Template 38	89.75	+1.75%
Template 39	83.64	-4.36%
Template 40	90.19	+2.19%
Template 41	88.43	+0.43%
Template 42	85.96	-2.04%
Template 43	88.50	+0.50%
Template 44	84.46	-3.54%
Template 45	86.25	-1.75%
Template 46	89.90	+1.90%
Template 47	89.87	+1.87%
Template 48	76.85	-11.15%
Template 49	85.63	-2.37%
Template 50	82.56	-5.44%

Table 17: Success rate and deviation from average for each template


```
            {
                "WORK_DESCRIPTION-
                    STR-2": "
                    WORK_DESCRIPTION
                    -2"
            }
        ]
    },
    {
        "{{START_DATE}}": "
            START_DATE",
        "{{END_DATE}}": "END_DATE",
        "{{JOB_TITLE}}": "JOB_TITLE"
            ,
        "{{COMPANY}}": "COMPANY",
        "{{LOCATION}}": "LOCATION",
        "{{WORK_DESCRIPTION-STR-
            MULTIPLE-3}}": [
            {
                "WORK_DESCRIPTION-
                    STR-0": "
                    WORK_DESCRIPTION
                    -0"
            },
            {
                "WORK_DESCRIPTION-
                    STR-1": "
                    WORK_DESCRIPTION
                    -1"
            },
            {
                "WORK_DESCRIPTION-
                    STR-2": "
                    WORK_DESCRIPTION
                    -2"
            }
        ]
    },
    {
        "{{START_DATE}}": "
            START_DATE",
        "{{END_DATE}}": "END_DATE",
        "{{JOB_TITLE}}": "JOB_TITLE"
            ,
        "{{COMPANY}}": "COMPANY",
        "{{LOCATION}}": "LOCATION",
        "{{WORK_DESCRIPTION-STR-
            MULTIPLE-3}}": [
            {
                "WORK_DESCRIPTION-
                    STR-0": "
                    WORK_DESCRIPTION
                    -0"
            },
            {
                "WORK_DESCRIPTION-
                    STR-1": "
                    WORK_DESCRIPTION
                    -1"
            },
            {
                "WORK_DESCRIPTION-
                    STR-2": "
                    WORK_DESCRIPTION
                    -2"
            }
        ]
    }
],
"Education": [
    {
        "{{GRADUATION_YEAR}}": "
            GRADUATION_YEAR",
        "{{DEGREE}}": "DEGREE",
        "{{INSTITUTION}}": "
            INSTITUTION",
        "{{LOCATION}}": "LOCATION"
    },
    {
        "{{GRADUATION_YEAR}}": "
            GRADUATION_YEAR",
        "{{DEGREE}}": "DEGREE",
        "{{INSTITUTION}}": "
            INSTITUTION",
        "{{LOCATION}}": "LOCATION"
    }
],
"field-skills": [
```

Model	German	English	Spanish	French	Chinese
CodeLlama-34b-Instruct-hf	0.8996	0.9562	0.8956	0.8892	0.8750
CodeLlama-34b-Instruct-hf_json_mode	0.8977	0.9512	0.8917	0.8856	0.8595
CodeLlama-7b-Instruct-hf	0.6713	0.8267	0.6706	0.6631	0.7056
CodeLlama-7b-Instruct-hf_json_mode	0.7583	0.8789	0.7176	0.6903	0.7392
DeepSeek-R1-Distill-Llama-8B	0.8811	0.9562	0.8907	0.8843	0.8851
DeepSeek-R1-Distill-Llama-8B_json_mode	0.8720	0.9508	0.8822	0.8762	0.8740
DeepSeek-R1-Distill-Qwen-1.5B	0.7204	0.9142	0.7319	0.7175	0.8204
DeepSeek-R1-Distill-Qwen-1.5B_json_mode	0.6874	0.9039	0.7124	0.6943	0.7985
DeepSeek-R1-Distill-Qwen-7B	0.8421	0.9498	0.8536	0.8489	0.8819
DeepSeek-R1-Distill-Qwen-7B_json_mode	0.8536	0.9468	0.8628	0.8659	0.8755
Falcon3-7B-Instruct	0.8895	0.9588	0.9023	0.8936	0.8681
Falcon3-7B-Instruct_json_mode	0.8872	0.9583	0.9030	0.8908	0.8665
Llama-3.1-70B-Instruct	0.9289	0.9701	0.9265	0.9180	0.8933
Llama-3.1-70B-Instruct_json_mode	0.9173	0.9604	0.9182	0.9146	0.8934
Llama-3.1-8B-Instruct	0.9154	0.9652	0.9160	0.9132	0.8742
Llama-3.1-8B-Instruct_json_mode	0.9004	0.9594	0.8992	0.8958	0.8768
Llama-3.2-1B-Instruct	0.7721	0.9083	0.7590	0.7503	0.7675
Llama-3.2-1B-Instruct_json_mode	0.7717	0.9092	0.7605	0.7604	0.7666
Llama-3.2-3B-Instruct	0.8476	0.9459	0.8728	0.8565	0.8598
Llama-3.2-3B-Instruct_json_mode	0.8034	0.9169	0.8137	0.7993	0.8288
Ministral-8B-Instruct-2410	0.8706	0.9330	0.8552	0.8546	0.8714
Ministral-8B-Instruct-2410_json_mode	0.8740	0.9343	0.8592	0.8619	0.8681
Mistral-Small-24B-Instruct-2501	0.9028	0.9425	0.9070	0.8879	0.8678
Mistral-Small-24B-Instruct-2501_json_mode	0.8864	0.9276	0.8864	0.8826	0.8654
Qwen2.5-0.5B-Instruct	0.7439	0.9092	0.7592	0.7387	0.8053
Qwen2.5-0.5B-Instruct_json_mode	0.7287	0.8971	0.7475	0.7267	0.8042
Qwen2.5-1.5B-Instruct	0.8211	0.9291	0.8353	0.8369	0.8611
Qwen2.5-1.5B-Instruct_json_mode	0.8100	0.9274	0.8296	0.8260	0.8546
Qwen2.5-14B-Instruct	0.9302	0.9702	0.9298	0.9220	0.9068
Qwen2.5-14B-Instruct_json_mode	0.9166	0.9643	0.9159	0.9097	0.9062
Qwen2.5-32B-Instruct	0.9030	0.9573	0.9121	0.9004	0.8876
Qwen2.5-32B-Instruct_json_mode	0.8697	0.9233	0.8765	0.8716	0.8858
Qwen2.5-3B-Instruct	0.8675	0.9376	0.8625	0.8596	0.8784
Qwen2.5-3B-Instruct_json_mode	0.8612	0.9298	0.8513	0.8491	0.8744
Qwen2.5-72B-Instruct	0.9300	0.9701	0.9211	0.9211	0.8992
Qwen2.5-72B-Instruct_json_mode	0.9064	0.9456	0.8950	0.8981	0.8963
Qwen2.5-7B-Instruct	0.8815	0.9269	0.8811	0.8711	0.8927
Qwen2.5-7B-Instruct_json_mode	0.8766	0.9278	0.8813	0.8694	0.8914
Qwen2.5-Coder-32B-Instruct	0.8912	0.9397	0.8936	0.8957	0.8951
Qwen2.5-Coder-32B-Instruct_json_mode	0.8859	0.9286	0.8864	0.8842	0.8956
Qwen2.5-Coder-7B-Instruct	0.8353	0.9118	0.8142	0.8217	0.8774
Qwen2.5-Coder-7B-Instruct_json_mode	0.8393	0.9154	0.8215	0.8295	0.8827
gemini-2.0-flash	0.9112	0.9272	0.9023	0.9125	0.8637
gemini-2.0-flash_json_mode	0.9165	0.9407	0.8864	0.8958	0.8573
gpt-4o-mini	0.9179	0.9388	0.9108	0.8904	0.8849
gpt-4o-mini_json_mode	0.9117	0.9427	0.9040	0.8979	0.8799
gpt-4o	0.9002	0.9626	0.8923	0.8893	0.8439
gpt-4o_json_mode	0.8881	0.9412	0.8875	0.8839	0.8629

Table 18: BERTScore Supplementary Materials: language-wise model performance on BERTScore

```
    {
        "{{SKILL_1}}": "SKILL_1",
        "{{SKILL_2}}": "SKILL_2",
        "{{SKILL_3}}": "SKILL_3",
        "{{SKILL_4}}": "SKILL_4"
    }
],
"field-language": [
```

```
{
    "{{LANGUAGE_1}}": "
        LANGUAGE_1",
    "{{LANGUAGE_1_LEVEL_TEXT}}":
         "LANGUAGE_1_LEVEL_TEXT"
        ,
    "{{LANGUAGE_2}}": "
        LANGUAGE_2",
```

Rank	Model	Average Time	Mode
1	gemini-2.0-flash	12.06s	Standard
2	gemini-2.0-flash	12.07s	JSON Mode
3	Llama-3.2-1B-Instruct	21.22s	Standard
4	Qwen2.5-7B-Instruct	22.97s	JSON Mode
5	gpt-4o-mini	23.88s	Standard
6	Qwen2.5-Coder-7B-Instruct	26.06s	JSON Mode
7	Qwen2.5-0.5B-Instruct	27.58s	Standard
8	gpt-4o-mini	27.83s	JSON Mode
9	Qwen2.5-1.5B-Instruct	29.30s	Standard
10	gpt-4o	30.94s	Standard
11	gpt-4o	31.19s	JSON Mode
12	Qwen2.5-0.5B-Instruct	41.96s	JSON Mode
13	Qwen2.5-3B-Instruct	53.10s	Standard
14	Llama-3.2-1B-Instruct	58.28s	JSON Mode
15	Llama-3.2-3B-Instruct	59.48s	Standard
16	Llama-3.1-8B-Instruct	67.08s (1.1m)	JSON Mode
17	Qwen2.5-3B-Instruct	67.41s (1.1m)	JSON Mode
18	Qwen2.5-7B-Instruct	70.33s (1.2m)	Standard
19	DeepSeek-R1-Distill-Qwen-1.5B	72.24s (1.2m)	Standard
20	Qwen2.5-1.5B-Instruct	81.85s (1.4m)	JSON Mode
21	Qwen2.5-Coder-7B-Instruct	85.58s (1.4m)	Standard
22	Falcon3-7B-Instruct	95.73s (1.6m)	Standard
23	DeepSeek-R1-Distill-Qwen-7B	107.97s (1.8m)	Standard
24	Llama-3.1-8B-Instruct	115.72s (1.9m)	Standard
25	Ministral-8B-Instruct-2410	125.10s (2.1m)	Standard
26	DeepSeek-R1-Distill-Qwen-1.5B	126.10s (2.1m)	JSON Mode
27	Falcon3-7B-Instruct	132.72s (2.2m)	JSON Mode
28	DeepSeek-R1-Distill-Llama-8B	136.10s (2.3m)	Standard
29	Qwen2.5-14B-Instruct	150.24s (2.5m)	Standard
30	DeepSeek-R1-Distill-Qwen-7B	165.65s (2.8m)	JSON Mode
31	CodeLlama-7b-Instruct-hf	165.74s (2.8m)	Standard
32	Mistral-Small-24B-Instruct-2501	166.72s (2.8m)	Standard
33	Mistral-Small-24B-Instruct-2501	169.09s (2.8m)	JSON Mode
34	Llama-3.2-3B-Instruct	186.67s (3.1m)	JSON Mode
35	Qwen2.5-14B-Instruct	201.00s (3.4m)	JSON Mode
36	Ministral-8B-Instruct-2410	223.77s (3.7m)	JSON Mode
37	Qwen2.5-32B-Instruct	228.36s (3.8m)	Standard
38	Qwen2.5-Coder-32B-Instruct	230.43s (3.8m)	JSON Mode
39	Qwen2.5-32B-Instruct	255.66s (4.3m)	JSON Mode
40	CodeLlama-7b-Instruct-hf	260.41s (4.3m)	JSON Mode
41	DeepSeek-R1-Distill-Llama-8B	287.35s (4.8m)	JSON Mode
42	Qwen2.5-72B-Instruct	418.83s (7.0m)	JSON Mode
43	CodeLlama-34b-Instruct-hf	421.07s (7.0m)	Standard
44	Qwen2.5-72B-Instruct	423.61s (7.1m)	Standard
45	Llama-3.1-70B-Instruct	439.67s (7.3m)	Standard
46	CodeLlama-34b-Instruct-hf	475.47s (7.9m)	JSON Mode
47	Llama-3.1-70B-Instruct	556.92s (9.3m)	JSON Mode
48	Qwen2.5-Coder-32B-Instruct	709.58s (11.8m)	Standard

Table 19: Average inference time by model and mode (lower is better)

```
            "{{LANGUAGE_2_LEVEL_TEXT}}":
                "LANGUAGE_2_LEVEL_TEXT"
        }
    ]
}
```

Parsing Schema for Resumebench-Mix (Most-Common Schema)

Listing 2: Parsing Schema for Resumebench-Mix (Most-Common Schema)

```
{
    "field-contact": {
        "{{NAME}}": "NAME",
        "{{PHONE_NUMBER}}": "
            PHONE_NUMBER",
        "{{EMAIL_ADDRESS}}": "
            EMAIL_ADDRESS"
    },
    "Work Experience": [
        {
            "{{COMPANY_NAME}}": "
                COMPANY_NAME",
            "{{JOB_TITLE}}": "JOB_TITLE"
                ,
            "{{START_TIME_TO_END_TIME}}"
                : "
                START_TIME_TO_END_TIME",
            "{{WORK_DESCRIPTION-STR-
                MULTIPLE}}": [
                {
                    "WORK_DESCRIPTION":
                        "
                        WORK_DESCRIPTION
                        "
                }
            ]
        }
    ],
    "Education": [
        {
            "{{INSTITUTION_NAME}}": "
                INSTITUTION_NAME",
            "{{DEGREE_NAME}}": "
                DEGREE_NAME",
            "{{START_TIME_AND_END_TIME}}
                ": "
                START_TIME_AND_END_TIME"
        }
    ]
}
```

N Prompts for Evaluation

We use the above prompt to guide the evaluation process, as we find that this prompt can effectively control various LLMs to generate the desired output.

```
Your task is to extract information from the
provided resume text into the provided **VALID
JSON format**.

The resume text is as follows: <resume_text>
<placeholder> </resume_text>

Instructions: 1. Do not change the content of
the resume text, you only need to extract the
required information. 2. Only output the valid
JSON format as shown below. 3. Do not modify
the key names in the JSON format. 4. **Do not
translate the language of the resume text.**
Your response should be structured in JSON format
as shown below: <JSON_FORMAT>
```

where we replace <placeholder> and <JSON_FORMAT> with the corresponding resume text and the appropriate parsing schema.

O Dataset Analysis Metrics for ResumeBench

O.1 Common Metrics

Refer to Table 20. The Kaggle dataset is the same as the Kaggle dataset in Section 3.3.

O.2 LLM-As-A-Judge

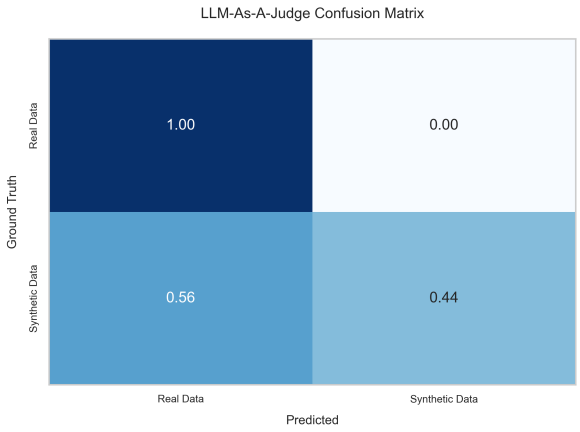


Figure 6: Confusion Matrix on LLM-As-A-Judge for real-or-synthetic resume binary classification from Resumebench-Mix

In our LLM-as-a-Judge experiment, we observe that ChatGPT-4o achieves 100% accuracy in evaluating real résumé samples. However, more than half of our synthetic samples are misclassified as real by ChatGPT-4o, demonstrating the effectiveness of our synthetic data in mimicking the characteristics of authentic resumes.

P Resume Samples

P.1 English

Please refer to Figure 7.

Dataset	Unigram TTR	Bigram TTR	NP Ratio	NP Length	Verb Ratio	Verb Subtree Length
Resumebench-EN	0.5898	0.9172	0.1163	3.8551	0.0511	26.1485
Resumebench-ZH	0.5376	0.8896	–	–	0.1168	13.2349
Resumebench-DE	0.6289	0.9181	0.1749	2.1611	0.0104	19.3595
Resumebench-FR	0.4861	0.8536	0.1638	2.3424	0.0322	23.2585
Resumebench-ES	0.4889	0.8420	0.1647	2.2781	0.0295	27.8825
Kaggle dataset (EN)	0.4757	0.8392	0.2050	2.5724	0.0874	21.1797
Resumebench-REAL-EN	0.5515	0.8933	0.1993	2.6915	0.0822	18.9395
Resumebench-REAL-ZH	0.5018	0.8494	–	–	0.1297	13.5563
Resumebench-REAL-DE	0.6670	0.9068	0.2215	2.0655	0.0260	16.7983
Resumebench-REAL-FR	0.5577	0.8260	0.2662	1.5107	0.0320	15.7689
Resumebench-REAL-ES	0.5057	0.7861	0.2167	2.4817	0.0257	19.9568

Table 20: Common Metrics on Resume Dataset

	English (en)	Chinese (zh)	French (fr)	Germany (de)	Spanish (es)	Total
Resumebench-Real	52	41	37	45	36	211
Resumebench-Mix	104	82	74	90	72	422

Table 21: ResumeBench-Mix Data Distribution

P.2 Chinese

Please refer to Figure 8.

P.3 Spanish

Please refer to Figure 9.

P.4 France

Please refer to Figure 10 and Figure 11.

P.5 German

Please refer to Figure 12 and Figure 13

Q ResumeBench-Mix

The sample distribution is presented in Table 21.

Figure 7: Resume Sample in English

张伟

法律顾问

个人简介

具有5年法律服务经验的专业法律顾问，擅长公司法、知识产权法及合同法。曾在知名律师事务所工作，积累了丰富的法律实务经验，能够为客户提供高效的法律支持和解决方案。

联系方式

邮箱: zhangwei@example.com

电话: 13800001234

个人信息

国籍: 中国，北京

语言: 中文，英文

能力

公司法 合同审查与管理 知识产权保护

工作经历

律师助理 at 北京金枫律师事务所

2016年6月 - 2018年4月

- 协助律师进行法律研究，整理和分析相关法律文献及案例，为律师提供全面的法律支持。
- 参与客户会议，记录讨论内容，并跟进案件进展，确保法律事务的高效处理与客户沟通顺畅。
- 支持律师团队的备案资料，确保案件资料的完整性与准确性，以提高案件处理效率。

法律顾问 at 环球律师事务所

2018年5月 - 至今

- 为国内外客户提供法律咨询服务，解答复杂的法律问题，并制订法律策略，以支持客户的商业决策。
- 负责起草、审核和修改各类法律文件及合同，确保其符合相关法律法规，并维护客户的合法权益。
- 参与并支持重大交易和诉讼案件的准备工作，协助客户制定应对方案，优化法律策略，以降低潜在风险。

教育背景

法律硕士

中山大学 (2014年9月 - 2016年6月)

专注于国际法与商法，参与多个课题研究，并撰写相关学术论文。在此期间，深化了对国际法理论和实际应用的理解，积极参与学校的法律实务每月沙龙，锻炼了法律分析和辩论能力。

法学学士

暨南大学 (2010年9月 - 2014年6月)

在校期间积极参与法律实践活动，参与多起模拟法庭的组织与执行，获得优秀法律实践奖。通过选修多门领域的课程，提高了对民法与商法的理解，打下了扎实的法学基础。

法学学士

浙江大学 (2010年9月 - 2014年6月)

获得法学专业知识的同时，参与了多项法律研究项目，锻炼了法律文书写作和案件分析能力，提升了团队合作能力。在校期间获得'优秀学生奖学金'，体现了课业表现与科研能力的结合。

证书

律师执业证书

北京市司法局

国际商法证书

国际商会（ICC）

兴趣爱好

- 阅读法律相关书籍
- 参加法律研讨会
- 旅行和探索不同文化

Figure 8: Resume Sample in Chinese

JUAN PÉREZ

PROFESIONAL DE MANUFACTURA Y PRODUCCIÓN

: 612-345-678

: juan.perez@email.com

: https://www.linkedin.com/in/juanperez

: Calle Falsa 123, Madrid, España

RESUMEN

Soy un profesional de manufactura y producción con más de 5 años de experiencia en la industria. He trabajado en entornos de alta presión, asegurando la calidad y eficiencia en los procesos de producción. Mi enfoque en el mantenimiento preventivo y correctivo ha contribuido a la reducción de costos y a la mejora de la productividad en mis anteriores empleos.

EXPERIENCIA LABORAL

Técnico de Mantenimiento

Mantenimiento de Maquinaria ABC | Enero 2015 – Abril 2017

- Efectué el mantenimiento preventivo de maquinaria industrial, asegurando el funcionamiento continuado en varias industrias como la de construcción y manufactura.
- Implementé un programa de mantenimiento correctivo que constató una mejora en la durabilidad de los equipos en un 10%.
- Asistí en la formación de un equipo nuevo de técnicos de mantenimiento, proporcionándoles herramientas y conocimientos para un desempeño efectivo.
- Colaboré en proyectos de mejora de procesos que involucraron la reducción de costos operativos mediante la optimización de rutinas de mantenimiento.

Técnico de Mantenimiento

Grupo Siro | Mayo 2017 – Diciembre 2018

- Llevé a cabo el mantenimiento diario de las líneas de producción, asegurando que todos los procesos se realizaban sin contratiempos en la producción alimentaria.
- Desarrollé procedimientos de mantenimiento que redujeron las averías de la maquinaria en un 15%, contribuyendo a un aumento significativo en la producción.
- Realicé diagnósticos técnicos para identificar fallos y proponer soluciones efectivas, mejorando el tiempo de respuesta en las reparaciones.
- Colaboré con otros departamentos para optimizar los procesos de producción y minimizar desperdicios, contribuyendo a iniciativas de sostenibilidad.

Técnico de Mantenimiento

Ferrovial | Enero 2019 – Actualidad

- Responsable del mantenimiento preventivo y correctivo de maquinaria pesada, asegurando su operatividad y minimizando tiempos de inactividad en las plantas de producción.
- Implementé un sistema de gestión de mantenimiento que mejoró la eficiencia operativa en un 20%, aumentando la disponibilidad de maquinaria.
- Colaboré con el departamento de calidad para garantizar que los estándares de producción se cumplieran a través del mantenimiento adecuado de los equipos.
- Entrené a nuevos técnicos en procedimientos de mantenimiento y seguridad, mejorando la formación del equipo.

EDUCACIÓN

Grado en Ingeniería Técnica Industrial

Universidad de Salamanca

2012 – 2016

HABILIDADES

- Mantenimiento Preventivo
- Gestión de Proyectos
- Análisis de Fallas
- Optimización de Procesos

Figure 9: Resume Sample in Spanish

RÉSUMÉ PROFESSIONNEL

{{SUMMARY}}

Skills

- Gestion de la chaîne d'approvisionnement automobile
- Analyse des données de production
- Management d'équipe
- Optimisation des processus
- Lean Manufacturing
- Planification stratégique

Insérer une image

EXPÉRIENCE PROFESSIONNELLE

Technicien de Production - Citroën

Juin 2016 - Février 2018

- Surveillance des lignes de production et ajustement des machines afin de maintenir un taux de conformité de 98%.
- Formation de 15 nouveaux employés aux procédures de production et aux normes de sécurité, ce qui a réduit le taux d'accidents de travail de 30%.
- Participation active aux projets d'amélioration des performances, conduisant à une réduction des déchets de 10% et des coûts de production.
- Collaboration avec les équipes qualité pour identifier et résoudre en temps réel les problèmes sur la chaîne de production.

Analyste de Données Automobiles - Panhard

Mars 2018 - Décembre 2019

- Analyse des données de production pour identifier les tendances, améliorant ainsi l'efficacité opérationnelle de 20%.
- Développement de rapports et de tableaux de bord interactifs pour la direction, facilitant des décisions stratégiques basées sur des données précises.
- Collaboration avec les équipes de développement produit pour intégrer des retours d'expérience basés sur les données consommées, aboutissant à l'optimisation des cycles de développement.
- Participation à des projets d'innovation, notamment le développement d'un modèle prédictif pour la maintenance préventive des équipements.

Manager de Production Automobile - Ligier

Janvier 2020 - Présent

- Responsable de la gestion de l'ensemble du processus de production, avec une attention particulière portée au respect des délais et des normes de qualité ISO 9001.
- Réduction des temps d'arrêt de production de 15% en mettant en œuvre des procédures d'amélioration continue et des méthodologies Lean.
- Collaboration étroite avec les équipes d'ingénierie et de logistique pour assurer une adaptation rapide aux fluctuations de la demande, garantissant ainsi une production fluide.
- Encadrement d'une équipe de 30 opérateurs, en assurant leur formation continue et leur développement professionnel.
- Implémentation de nouvelles technologies de capteurs IoT pour optimiser la surveillance des lignes de production.

HISTOIRE DE L'ÉDUCATION

Master en Génie Mécanique et Production - Sorbonne Université

Septembre 2014 - Juin 2016

Formé aux techniques avancées de gestion de la production et à l'analyse des données dans le secteur automobile. A participé à des projets de recherche sur l'optimisation des

Figure 10: Resume Sample in French Page 1

lignes de production et sur l'intégration des nouvelles technologies telles que l'IoT dans l'industrie automobile, ce qui a renforcé mes compétences en gestion de projet et en utilisation de logiciels de simulation.

Licence en Sciences de l'Ingénieur - Université de Savoie Mont Blanc

Septembre 2011 - Juin 2014

Acquisition de compétences fondamentales en ingénierie, incluant des projets pratiques dans le domaine de l'automobile. Réalisation d'un projet de fin d'études sur le design et l'optimisation d'un prototype de véhicule électrique, impliquant des études de faisabilité et des tests de performance.

RÉCOMPENSES ET ACTIVITÉS

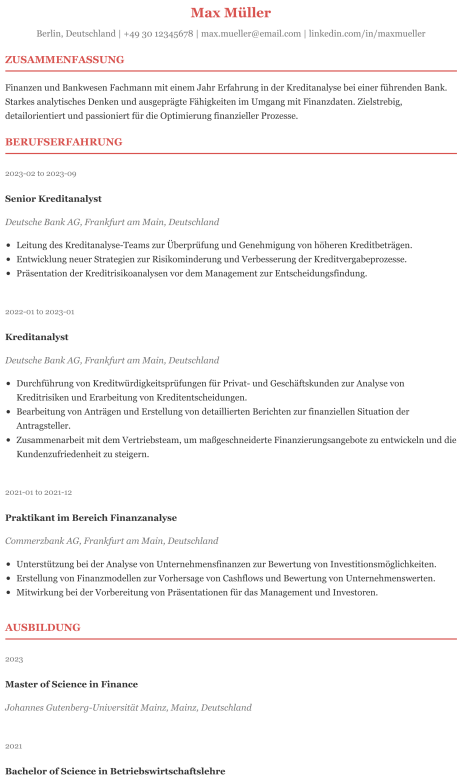
- Prix de l'Innovation en Production - Ligier 2021
- Meilleur Projet de Fin d'Études - Sorbonne Université 2016

Jean Dupont

+33 6 12 34 56 78

jean.dupont@example.com

Figure 11: Resume Sample in French Page 2



Max Müller

Berlin, Deutschland | +49 30 12345678 | max.mueller@email.com | linkedin.com/in/maxmueller

ZUSAMMENFASSUNG

Finanzen und Bankwesen Fachmann mit einem Jahr Erfahrung in der Kreditanalyse bei einer führenden Bank. Starkes analytisches Denken und ausgeprägte Fähigkeiten im Umgang mit Finanzdaten. Zielstrebig, detailorientiert und passioniert für die Optimierung finanzieller Prozesse.

BERUFSERFAHRUNG

2023-02 to 2023-09

Senior Kreditanalyst

Deutsche Bank AG, Frankfurt am Main, Deutschland

- Leitung des Kreditanalyse-Teams zur Überprüfung und Genehmigung von höheren Kreditbeträgen.
- Entwicklung neuer Strategien zur Risikominderung und Verbesserung der Kreditvergabeprozesse.
- Präsentation der Kreditrisikoanalysen vor dem Management zur Entscheidungsfindung.

2022-01 to 2023-01

Kreditanalyst

Deutsche Bank AG, Frankfurt am Main, Deutschland

- Durchführung von Kreditwürdigkeitsprüfungen für Privat- und Geschäftskunden zur Analyse von Kreditrisiken und Erarbeitung von Kreditentscheidungen.
- Bearbeitung von Anträgen und Erstellung von detaillierten Berichten zur finanziellen Situation der Antragsteller.
- Zusammenarbeit mit dem Vertriebsteam, um maßgeschneiderte Finanzierungsangebote zu entwickeln und die Kundenzufriedenheit zu steigern.

2021-01 to 2021-12

Praktikant im Bereich Finanzanalyse

Commerzbank AG, Frankfurt am Main, Deutschland

- Unterstützung bei der Analyse von Unternehmensfinanzen zur Bewertung von Investitionsmöglichkeiten.
- Erstellung von Finanzmodellen zur Vorhersage von Cashflows und Bewertung von Unternehmenswerten.
- Mitwirkung bei der Vorbereitung von Präsentationen für das Management und Investoren.

AUSBILDUNG

2023

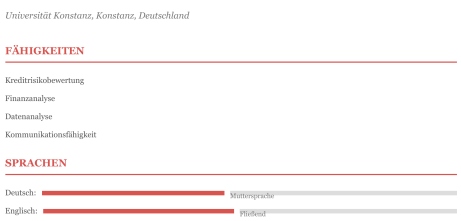
Master of Science in Finance

Johannes Gutenberg-Universität Mainz, Mainz, Deutschland

2021

Bachelor of Science in Betriebswirtschaftslehre

Figure 12: Resume Sample in German Page 1



Universität Konstanz, Konstanz, Deutschland

FÄHIGKEITEN

Kreditrisikobewertung

Finanzanalyse

Datenanalyse

Kommunikationsfähigkeit

SPRACHEN

Deutsch: Muttersprache

Englisch: Fließend

Figure 13: Resume Sample in German Page 2