Multi-Stage Job Advertisement Analysis

Automated Skill Extraction using BERT and Large Language Models

Project Report

Author: André Kuhn

Hochschule Luzern
Master of Science in Applied Information and Data Science (MSc IDS)

Module: Advanced Generative AI

December 23, 2025

Abstract

Processing unstructured job advertisements to identify specific professional requirements remains a challenge in recruitment analytics. This project addresses this issue by developing a hybrid Natural Language Processing (NLP) pipeline for automated skill extraction. The methodology employs a two-stage approach: first, a fine-tuned BERT model segments the text to locate relevant content; second, Google's Gemini Large Language Model (LLM) acts as a semantic filter to extract precise entities. Experimental results demonstrate that while the BERT model achieves a weighted F1-score of 0.84, the specific detection of skill zones proved difficult, characterized by low precision. The integration of the LLM compensates for this by refining the noisy output, mitigating ambiguity and producing standardized skill lists from multi-lingual text.

Contents

1 Introduction

1.1 Context and Motivation

The digital labor market has expanded substantially, distributing job advertisements across numerous platforms. However, the utility of this data is often constrained by its unstructured format. Job ads frequently lack a standardized schema, making it difficult for automated systems to distinguish between a "required skill" and a "benefit." Manual processing can be resource-intensive and difficult to scale. Consequently, there is a need for intelligent systems capable of parsing natural language to extract structured entities like professional skills, bridging the gap between raw text and actionable insights [1].

1.2 Problem Statement

A central challenge addressed by this project is the semantic gap between unstructured recruitment text and structured data systems. Traditional keyword-based algorithms often struggle to distinguish the context of a word. To bridge this gap, this project aims to develop an end-to-end analytical pipeline designed to transform raw job advertisement text into structured insights by addressing two sequential challenges:

1. **Automated Zone Identification:** The first challenge is to dissect job ads into meaningful sections, separating relevant requirements from extraneous information like legal disclaimers. The objective is to address this by training a discriminative deep learning model (BERT) [2] on a labeled dataset to segment the text into functional zones such as "Fähigkeiten und Inhalte" (skills).

2. **Targeted Skill Extraction:** The second challenge is to convert the natural language text within those isolated zones into machine-readable data. The goal is to leverage the semantic reasoning capabilities of State-of-the-art Large Language Models (LLMs) to extract specific professional skills.

2 Dataset Descriptions

2.1 Initial Annotated Dataset

The primary resource for this project was the `annotated.json` dataset provided. This dataset consists of 2,699 job advertisements, predominantly in German, with some instances in French and English. The documents were manually annotated with character-level offsets mapping to specific semantic zones. The annotation schema includes labels such as:

- *Fähigkeiten und Inhalte* (Skills and Content - Target Class)
- *Bewerbungsprozess* (Application Process)
- *Anstellung* (Employment Details)
- *Benefits* (Perks)
- *Challenges* (Tasks/Challenges)

A notable characteristic of this dataset is the class imbalance (Table 1). While "Fähigkeiten und Inhalte" and "Bewerbungsprozess" are well-represented, classes like "Challenges" and "Arbeitsumfeld" are rare, posing a potential challenge for supervised learning.

Label Class	**Document Count**	**Total Tokens**
Fähigkeiten und Inhalte (Target)	2,424	418,800
Bewerbungsprozess	2,052	188,370
Anstellung	2,393	119,603
Firmenbeschreibung	1,123	117,195
Benefits	1,280	104,506
Challenges	101	2,260
Arbeitsumfeld	55	2,102

Table 1: Label Distribution showing significant class imbalance.

2.2 Crawled Dataset

The project specifications included an optional task to crawl additional job advertisements from the web. For this specific project iteration, no external crawled dataset was utilized.

3 Methodology

The methodology followed a structured pipeline approach: Data Preparation, Discriminative Modeling (BERT), and Generative Extraction (LLM).

3.1 System Architecture

The system architecture is visualized in Figure 1.

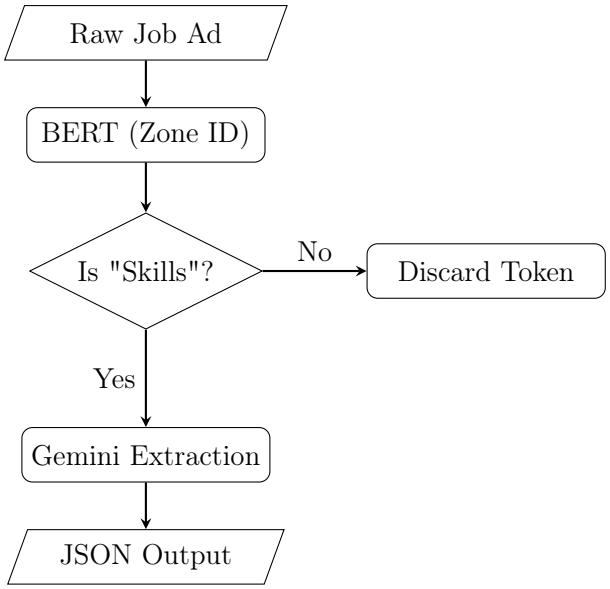

Figure 1: System Architecture Pipeline

3.2 Preprocessing and Tokenization

3.2.1 Tokenization and Alignment

BERT does not interpret raw strings; it requires numerical input. I utilized the `BertTokenizerFast` to tokenize the text. The preprocessing script aligned the character-level annotations provided in the JSON with the sub-word tokens generated by BERT.

For example, the word "TensorFlow" might be split into tokens "Tensor" and "##Flow". If the JSON annotation marked "TensorFlow" as a skill, the implemented logic was designed to ensure that both "Tensor" and "##Flow" received the corresponding label ID. This facilitates the model's ability to learn the full representation of the entity.

3.2.2 Sliding Window Technique

A constraint of BERT models is the maximum input sequence length of 512 tokens. Job advertisements, however, frequently exceed this length. Simply truncating the text could result in the loss of relevant information (e.g., skills listed at the end of the document).

To address this, a sliding window technique was implemented:

- **Window Size:** 512 tokens.
- **Stride:** 128 tokens.

This means the text was split into overlapping chunks. A long document of 1000 tokens would be split into roughly three chunks, with the end of the first chunk overlapping with the beginning of the second. This overlap aims to minimize the loss of semantic context at the cutting points. The final dataset expanded from 2,699 documents to 3,718 processable chunks.

3.3 Task 2: Zone Identification Model (BERT)

I instantiated a `BertForTokenClassification` model. The training loop was implemented in PyTorch with the following key components:

- **Optimizer:** AdamW (Adam with Weight Decay) was selected as it is a standard optimizer for Transformer models.
- **Scheduler:** A linear learning rate scheduler with warmup was used. This is intended to slowly increase the learning rate at the start of training to stabilize the weights before decaying it, helping to prevent the model from oscillating around local minima.
- **Loss Function:** CrossEntropyLoss was used. Measures were taken to ensure that padding tokens (used to make batches equal length) were ignored during loss calculation.

3.4 Task 5: Hybrid Skill Extraction Pipeline

The final stage involved integrating the trained BERT model with the Google Gemini API [3].

The Pipeline Logic:

1. **Inference:** The BERT model scans the document and identifies tokens predicted to belong to the *Fähigkeiten und Inhalte* (skills) zone.
2. **Reconstruction:** These tokens are concatenated back into a single text string. This is intended to remove non-skill text.
3. **Generative Extraction:** This filtered text is sent to Google Gemini 2.5 Flash with a specific prompt.

Prompt Engineering: I engineered a specific system prompt to enforce constraints:

> *"You are an expert HR analyst. Extract all professional skills (hard and soft) from the text. Constraints: Output strictly a JSON list of strings. Phrases 2-5 words long."*

4 Experimental Setup

The experiments were conducted on a high-performance workstation.

4.1 Hardware Environment

- **Processor:** 13th Gen Intel(R) Core(TM) i9-13900KF (3.00 GHz)
- **RAM:** 32.0 GB (DDR5)
- **GPU:** NVIDIA GeForce RTX 4090 (24 GB VRAM)
- **OS:** Windows 11 (64-bit)

The NVIDIA RTX 4090 is a consumer-grade flagship GPU. Its 24GB of VRAM allowed for a batch size of 8 without memory overflow, which is relatively high for BERT-based training.

4.2 Software Environment

The following key software libraries and versions were used:

- **Language:** Python 3.10
- **Core Framework:** PyTorch 2.x (with CUDA 12.x support)
- **Model Library:** Hugging Face Transformers 4.x
- **LLM Integration:** `google-genai` SDK (latest version supporting Gemini 2.x)
- **Evaluation:** Seqeval [4] (for IOB metric calculation)

4.3 Hyperparameters

The following hyperparameters were selected for fine-tuning BERT:

Parameter	**Value**
Model Architecture	`bert-base-multilingual-cased`
Epochs	5
Batch Size	8 (simulated via Gradient Accumulation)
Learning Rate	2×10^{-5}
Max Sequence Length	512
Optimizer	AdamW
Gradient Accumulation Steps	2

Table 2: Hyperparameter Configuration

5 Results and Analysis

5.1 Quantitative Evaluation (BERT)

The BERT model showed evidence of convergence, with training loss decreasing from 242.98 (Epoch 1) to 90.19 (Epoch 5). The final model achieved a weighted F1-Score of 0.84 on the test set. However, the aggregate score masks disparities between classes.

Class	Precision	Recall	F1-Score	Support
Benefits	0.86	0.85	0.86	21,727
Bewerbungsprozess	0.86	0.88	0.87	38,772
Erfahrung	0.78	0.82	0.80	7,678
Fähigkeiten und Inhalte	**0.29**	**0.46**	**0.36**	**1,157**
Challenges	0.00	0.00	0.00	31
Weighted Avg	**0.83**	**0.85**	**0.84**	**71,139**

Table 3: Detailed Classification Report. Note the lower precision for the target "Skills" class.

Interpretation: The model performed effectively on structural zones like "Application Process" (F1: 0.87) and "Benefits" (F1: 0.86). However, the performance on the target zone "Skills" was suboptimal (F1: 0.36). The low precision (0.29) indicates that the model appears to be "over-predicting" skills, identifying text as a skill when it likely belongs to another category. Furthermore, rare classes like "Challenges" were not successfully identified (F1: 0.00), which is likely a direct consequence of the data imbalance.

5.2 Qualitative Error Analysis

To investigate the lower precision in the target "Skills" zone, I performed a qualitative inspection of the model's errors on the test set.

5.2.1 Statistical Confusion Patterns

The confusion matrix (Table 4) reveals distinct error patterns. The most frequent error type involves the "Outside" class (irrelevant text), labeled as O.

- **Observation 1: Over-Prediction (False Positives).** A significant error involves false positives where the model incorrectly predicts "Fähigkeiten und Inhalte" (skills) instead of the true "Outside" (O) label (4,037 cases). This suggests that the model is sensitive, possibly associating general business vocabulary (e.g., "team," "tasks," "work") with skills, even when they appear in conversational text.

	Predicted Class				
True Class	O	Bewerb.	Firmen.	Anstell.	Fähigk.
O	**28802**	2482	1326	1235	**4037**
Bewerbung.	1868	**34024**	827	28	512
Firmenbes.	847	1181	**18270**	0	**1568**
Anstellung	785	709	562	**19378**	1050
Fähigkeiten	1987	470	1168	1252	**75421**

Table 4: Condensed Confusion Matrix. Diagonal values (bold) indicate True Positives.

- **Observation 2: Semantic Confusion with Company Description.** There were 1,568 instances where text describing the company was misclassified as a required skill.

5.2.2 Visual Inspection of Specific Errors

By examining specific failure cases in the test set, I identified possible causes for the model's errors.

1. **Job Title vs. Qualification Ambiguity** A frequent error appears to occur when a word can serve as both a job title and a qualification, creating semantic ambiguity.
    - *Example 1:* "Dafür suchen wir: Juristinnen und Juristen..."
    - *True Label:* Anstellung (Employment/Job Title)
    - *Predicted:* Abschlüsse (Qualifications)
    - *Analysis:* The term "Juristinnen/Jurist" (Lawyer) functions as an academic title. This prediction suggests the model may have internalized a strong association between academic titles and the Qualifications zone. In this instance, it appears to have struggled to recognize that the specific sentence structure ("We are looking for...") contextualizes the term as a Job Title rather than a prerequisite qualification.
2. **Rare Class Failure (Data Sparsity)** Classes with very few training examples seemed to be systematically ignored during inference.
    - *Example 2:* "...Begleitung von Beschaffungsprojekten **Pikettdienst**..."
    - *True Label:* Challenges
    - *Predicted:* Abschlüsse
    - *Analysis:* The label Challenges accounts for less than 0.5% of the dataset. The incorrect prediction implies that the model likely had insufficient exposure to vocabulary such as "Pikettdienst" (on-call duty) within this specific context.

3. **Cross-Lingual Structure Transfer** There are indications that the multilingual model may occasionally apply structural logic derived from German texts to French documents.

    - *Example 3:* "Votre **profil**: Vous disposez..."
    - *True Label:* Fähigkeiten und Inhalte
    - *Predicted:* Abschlüsse
    - *Analysis:* In German job advertisements, the header "Profil" is frequently followed by a list of formal degrees. The misclassification here suggests a possible negative transfer, where the model may have erroneously applied this learned German pattern to the French header "Votre profil," predicting the subsequent text as qualifications rather than general skills.

5.3 Gemini Integration

Despite the noise in the BERT output, the final pipeline results appeared effective. The LLM served as a semantic filter. The following case studies illustrate the raw input versus the cleaned output.

1. **Job ID 1: Multilingual Extraction (French)**

    Input (BERT): *"MISSION Effectuer les contrôles techniques conformément à la loi fédérale sur la circulation routière et à ses dispositions d ' exécution ainsi qu ' aux directives nationales..."*

    Output (Gemini):

    - "Contrôles techniques"
    - "Examens de conduite"
    - "Maîtrise bureautique"
    - "Connaissances informatiques"

    Observation: The pipeline processed French text without requiring specific language configuration or retraining.

2. **Job ID 2: German Security**

    Input (BERT): *"Ihre Aufgaben Sie leisten einerseits Dienste in der Zentrale (Überwachungsaufgaben , Bedienung Leitsystem , Administration , Kontaktstelle für Externe , etc .)..."*

    Output (Gemini):

    - "Bedienung Leitsystem"

- "Hohe Loyalität"
- "Hohe Belastbarkeit"
- "Affinität für sicherheitstechnische Systeme"

Observation: Gemini successfully extracted relevant skills, demonstrating semantic understanding of the context.

6 Challenges and Mitigation Strategies

Throughout the project, several technical challenges were encountered and addressed.

6.1 API Deprecation

The initial implementation using the legacy `google.generativeai` library failed due to model deprecation ('404 Not Found'). **Mitigation:** The system was updated to the modern `google-genai` SDK (v0.3+) and a dynamic `get_available_model()` function was implemented. This script programmatically queries the API for active models (e.g., `gemini-2.5-flash`) and selects the newest version.

6.2 Sliding Window Context Loss

Splitting documents into chunks carries the risk of "fracturing" entities (e.g., splitting "Machine Learning Engineer" across two windows), potentially degrading semantic context. **Mitigation:** A stride of 128 tokens was implemented. This overlap ensures that an entity cut at the edge of one chunk appears fully formed in the center of the adjacent chunk, facilitating correct capture during inference.

7 Conclusion and Future Work

This project proposed a multi-stage pipeline for automated skill extraction, designed to bridge the gap between unstructured job advertisements and structured data. The fine-tuned BERT model acted as a coarse filter. While it struggled with precise semantic boundaries, evidenced by its low precision (0.29), it achieved high recall, ensuring that the majority of skill-relevant text was captured. The subsequent integration of the LLM served as a refinement step, effectively filtering noisy output from the initial stage to enhance precision and facilitate data standardization.

7.1 Future Work

To advance the system beyond a proof-of-concept toward a more robust analytical tool, several potential lines of investigation are proposed. A primary area for exploration involves the standardization of extracted data. Currently, the system produces free-text strings, resulting in variations such as "Python programming" versus "Coding in Python." Additionally, the underlying model architecture might be optimized to further support multilingual processing. It is hypothesized that replacing the standard BERT model with XLM-RoBERTa could yield performance gains, particularly regarding cross-lingual transfer between German, French, and English advertisements, by leveraging its extensive pre-training on diverse multilingual corpora.

References

[1] Blattner, M. (2025). *Project Specification: Multi-Stage Job Advertisement Analysis using BERT and Large Language Models*. Lucerne University of Applied Sciences and Arts.

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805

[3] Google DeepMind. (2023). *Gemini: A Family of Highly Capable Multimodal Models*. Google Technical Report. https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf

[4] Nakayama, H. (2018). *seqeval: A Python framework for sequence labeling evaluation*. GitHub Repository. https://github.com/chakki-works/seqeval