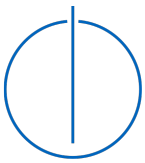
DEPARTMENT OF COMPUTER SCIENCE

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis

Gradient-Free Optimization of Retrieval Augmented Prompts for Legality Classification of Clauses in German Employment Contracts

David Pauschert



DEPARTMENT OF COMPUTER SCIENCE

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis

Gradient-Free Optimization of Retrieval Augmented Prompts for Legality Classification of Clauses in German Employment Contracts

Gradientenfreie Optimierung von wissensangereicherten Prompts zur Legalistätsklassifikation von Klauseln aus deutschen Arbeitsverträgen

Author:	David Pauschert
Supervisor:	Oliver Wardas, M.Sc.
Advisor:	Prof. Florian Matthes, Ph.D.
Submission Date:	26.02.2025

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 26.02.2025 David Pauschert

AI Assistant Usage Disclosure

Introduction

Performing work or conducting research at the Chair of Software Engineering for Business Information Systems (sebis) at TUM often entails dynamic and multi-faceted tasks. At sebis, we promote the responsible use of *AI Assistants* in the effective and efficient completion of such work. However, in the spirit of ethical and transparent research, we require all student researchers working with sebis to disclose their usage of such assistants.
For examples of correct and incorrect AI Assistant usage, please refer to the original, unabridged version of this form, located at this link.

Use of *AI Assistants* for Research Purposes

I have used AI Assistant(s) for the purposes of my research as part of this thesis.

Yes [x] No [ ]

Explanation: I have used ChatGPT, DeepL, and Grammarly for translations and stylistic refinement of texts previously drafted by the author.

I confirm in signing below, that I have reported all usage of AI Assistants for my research, and that the report is truthful and complete.

Munich. 26.02.2025	David Pauschert
Location, Date	Author

Abstract

Automated employment contract review is among the abundance of promising applications of Large Language Models (LLMs) in the legal domain. Employment contracts are designed to protect both employers and employees, yet evolving legislation, standardized templates, and individual modifications can introduce legally void clauses. Undetected, these clauses pose financial and reputational risks for employers and may weaken employees' contractual protections. As a result, contract reviews are often outsourced to law firms, a process that is both costly and time-consuming. An automated, LLM-driven fairness and legality assessment of German employment contracts thus has the potential to streamline reviews, reduce costs, and enhance compliance, ultimately strengthening employees' legal standing. However, accurately classifying the fairness of contractual clauses requires expert knowledge in labor law, legal precedents, and court rulings—expertise that off-the-shelf LLMs typically lack. To bridge this knowledge gap, Retrieval-Augmented Generation (RAG) or Fine-Tuning can be employed. Yet, RAG depends on high-quality, consistent, and up-to-date legal data, which is often unavailable, while Fine-Tuning is resource-intensive and technically complex, positioning Prompt Engineering as a more accessible and scalable approach. However, manual prompt engineering is labor-intensive and time-consuming, which is why Automated Prompt Optimization (APO) has emerged as a promising solution. Even so, existing APO applications have primarily focused on enhancing instruction-following behavior, leaving open the question of whether APO can implicitly acquire the legal knowledge necessary for fairness classification of clauses in German employment contracts. To address this gap, we develop a novel APO algorithm based on beam search, combining global exploration and local exploitation to optimize category-specific prompts. We evaluate the performance of APO-optimized prompts against a domain-expert-written baseline on a dataset of German employment contract clauses. Our results show that APO yields only a marginal 2-percentage-point increase in global F1 but improves the detection of void clauses by 45%—a significant success given their legal and financial implications. We hypothesize that the limited performance gains stem from algorithmic overfitting, dataset size constraints, class imbalance, and potential mismatches between LLM scale and task complexity. Our findings highlight key challenges in applying APO to downstreams tasks in knowledge-intensive domains such as law and medicine.

Contents

1. Introduction

This chapter serves as the foundation of this thesis. We introduce the topic, identify the research gap and corresponding research questions, outline our methodological contributions in addressing these questions, review related work relevant to our study, and provide a comprehensive overview of the thesis structure and objectives.

1.1. Motivation

Large Language Models (LLMs) have achieved remarkable success across various natural language processing tasks, demonstrating their ability to generate, summarize, and analyze text with human-like fluency [1, 2]. One domain that stands to benefit significantly from this automation potential is the legal sector, where tasks such as contract analysis, legal document drafting, and compliance checks traditionally require extensive human expertise [3]. The structured yet ambiguous nature of legal language makes these processes time-consuming and prone to inconsistencies. LLMs offer the possibility of improving efficiency, accuracy, and fairness in legal decision-making [4].

Applications of LLMs in the legal field range from legal case retrieval [5], legal question-answering [6], and judgment prediction [7, 8] to document drafting [9] and semantic annotations [10]. Another area where LLMs hold significant transformative potential is contract analysis – In this context, the KIBekodA project [11] aims to design, implement, and evaluate a prototypical LLM-based system for assisting legal professionals in the review and correction of German employment contracts as part of a collaborative effort. The system classifies contract clauses based on fairness and suggests replacements for unfair or void clauses to ensure compliance with German law, thereby increasing efficiency and potentially improving accuracy.

Despite these promising applications, the deployment of LLMs in the legal domain presents significant challenges [12]. Legal texts are highly structured, incorporate specialized terminology, and often allow for multiple interpretations, making it difficult for LLMs to consistently provide accurate and contextually appropriate responses [13]. Off-the-shelf models struggle with the complexities of legal reasoning and frequently misinterpret statutes or legal language [14]. Moreover, LLMs are prone to hallucination, sometimes fabricating citations or generating incorrect legal conclusions, which is particularly problematic in a field where precision is critical [15].

These challenges have driven research toward three key methods for adapting LLMs to legal tasks: fine-tuning specialized models [16, 17], retrieval-augmented generation (RAG) [18, 19, 20], and prompt engineering [21, 22]. Since fine-tuning is resource-intensive and technically complex, and retrieval-augmented generation (RAG) relies on access to high-quality, consistent, up-to-date legal data sources that are often unavailable, prompt engineering can be viewed as the most accessible and practical approach for enhancing LLM performance on downstream legal tasks. While prompt engineering offers a practical alternative to fine-tuning and retrieval-augmented generation, it comes with its own challenges. Manually crafting effective prompts is a laborious and time-consuming process, requiring domain experts to iteratively refine prompts for optimal performance [23]. Moreover, manual approaches do not guarantee consistent improvements, as small variations in phrasing can lead to unpredictable model behavior. Given these limitations, automated prompt optimization (APO) has emerged as a promising solution to systematically discover, evaluate, and refine prompts without human intervention [24].

APO aims to automate the search for the most effective prompts for a given downstream LLM task, eliminating the inefficiencies of manual prompt engineering [25]. Early research has explored techniques such as soft prompt tuning [26, 27, 28, 29, 30], reinforcement learning [31, 32], and Bayesian optimization [33, 34, 35], yet these methods often require direct access to model parameters and gradients. As modern LLMs increasingly operate as black-box systems with API-only access, these approaches become infeasible. This has led to the development of LLM-based APO, where the model itself assists in optimizing its prompts through iterative refinement. By leveraging techniques such as evolutionary algorithms [36, 37], self-reflection [38, 39], and black-box optimization [40], these methods have demonstrated success on various benchmark datasets across a variety of tasks.

1.2. Problem Statement

Despite their obvious success, APO algorithms are primarily developed to refine the instructional component of prompts that guides the model on how to approach a specific task [25]. However, for many downstream LLM applications in the legal domain, such as judgment prediction, the task formulation is relatively well-defined. In contrast, tasks like legal clause classification require LLMs to have legal knowledge about economic law, legal precedents, and court rulings to accurately assess the fairness of a contractual clauses. This knowledge could potentially be provided in the form of finetuning or RAG which come with the aforementioned significant limitations. Given these constraints, a key question arises: Can we bridge the knowledge gap—moving

from off-the-shelf models to expert-level legal reasoning—by automatically deriving and eliciting the necessary legal knowledge through Automated Prompt Optimization?

To explore this, we focus on a concrete use case within the previously discussed KIBekodA project [11]. Specifically, we employ APO to optimize prompts for classifying fairness in contract clauses from the dataset in [41]. This dataset consists of 893 German-language contract clauses, each labeled as fair, unfair, or void, labeled by lawyers specialized in economic law. Since identifying and reviewing contract clauses for legal validity requires deep expertise in economic and labor law, this presents a challenging test case for evaluating APO's potential in augmenting off-the-shelf LLMs with expert knowledge.

In this context, we aim to answer the following research questions:

- **RQ1:** What is the current state of Automatic Prompt Optimization, and to what extent do existing approaches consider expert knowledge in prompt optimization?
- **RQ2:** How can existing methods be adapted to design an APO algorithm that optimizes prompts used for classifying clauses from German contracts by implicity learning the required legal expert knowledge?
- **RQ3:** How does this new APO algorithm perform in classifying the fairness of clauses from German employment contracts compared to an unoptimized expert-written prompt as a benchmark using the Wardas-Matthes [41] dataset?

Addressing these research questions is relevant from both practical and theoretical perspectives. Practically, improving the classification of unfair or void clauses contributes to ensuring fair contract conditions for employees while reducing legal risks for employers. Theoretically, this research extends prompt engineering methodologies for knowledge-intensive domains, such as law and medicine, where structured domain expertise is essential for accurate and reliable LLM outputs.

1.3. Contribution

To answer the research questions, we employ a combination of literature review, algorithm development, and experimental benchmarking.

To address the first research question (**RQ 1**), we conduct a comprehensive literature review on Automatic Prompt Engineering (APO), outlining its current state and methodological advancements. We analyze two primary paradigms within APO: Instruction Optimization (IO) and Exemplar Optimization (EO) [25]. Additionally, we examine how traditional approaches such as Soft Prompt Tuning, Reinforcement Learning, and Bayesian Optimization have been increasingly replaced by LLM-based

prompt optimization techniques. We then delve deeper into LLM-based APO methods by abstracting individual approaches from the literature into a generalizable framework and developing a taxonomy of methodologies to expand the prompt search space.

To answer the second research question (**RQ 2**), we develop a new APO algorithm that integrates several concepts derived from the literature. The proposed method is built on an iterative optimization process, incorporating Beam Search for efficient prompt selection and refinement. Furthermore, it leverages LLM self-reflection capabilities and evolutionary algorithms as a means to expand the prompt search space in each iteration. Inspired by divide-and-conquer principles in prompt tuning [42], the approach does not rely on a single global prompt but instead optimizes prompts for distinct thematic categories within employment contracts.

Finally, to address the third research question (**RQ 3**), we evaluate the performance of the developed APO algorithm through benchmarking experiments. The evaluation compares the effectiveness of category-specific APO-optimized prompts against a category-agnostic, expert-written prompt on the aforementioned dataset of contract clauses [41]. In addition to standard performance metrics such as Accuracy, F1-Score, Precision, and Recall, we analyze the optimization trajectory of the APO algorithm to assess its impact on performance improvement over the optimization course.

1.4. Related Work

To the best of our knowledge, no prior work has applied automatic prompt optimization to any downstream LLM task in the legal domain, let alone to acquiring the necessary legal knowledge for accurate task performance. Methodologically, our research aligns with studies in **Automatic Prompt Optimization (APO)**, while in terms of application, it broadly relates to research on **Large Language Models in the Legal Domain** and more specifically to **Contract Analysis**.

Automatic Prompt Optimization (APO)

The origins of Automatic Prompt Optimization (APO) can be traced back to Soft Prompt Tuning [26, 27, 28, 29, 30]. However, as this approach often produces uninterpretable prompts, subsequent research has explored reinforcement learning [31, 32] and Bayesian optimization [33, 34, 35] as alternative tuning methods. Despite these advancements, such techniques still require parameter or gradient-level access, making them increasingly impractical as modern LLMs transition towards black-box, API-only access. This shift has driven the adoption of LLM-based APO methods, where the model itself handles the entire prompt optimization process, from discovery to evaluation and selection. Within this paradigm, existing methods can be classified into resampling-based

[24], reflection-based [38, 39], evolutionary algorithm-driven [36, 37], and black-box optimization approaches [40].

Large Language Models in the Legal Domain

Several surveys have been conducted that provide a detailed overview of use cases for applying large language models in the legal domain [3, 43]. Application areas of LLMs in the legal domain range from legal judgment prediction and reasoning [21, 22, 44, 45, 46, 47] over legal information extraction and summarization [48, 49, 50] to LLM-based legal advisory and AI-assisted lawyering [51, 52].

AI-Driven Contract Analysis

The most relevant work to ours is by Wardas and Matthes [41], who introduce a dataset of annotated contractual clauses from German employment contracts, which serves as the training data for our APO algorithm. Lee et al. [53] propose an automatic contract-risk extraction model based on natural language processing (NLP) to detect "poisonous clauses" in construction contracts, focusing on deviations from the FIDIC Red and Pink Book standards that unfairly shift risk to contractors. Hendrycks et al. [54] present CUAD, a large-scale contract review dataset with expert annotations, designed to assess the ability of NLP models to generalize to legal tasks and to facilitate contract analysis automation. Leivaditi et al. [55] introduce ALeaseBERT, a language model tailored for lease agreement analysis, accompanied by a newly annotated dataset of 179 lease contracts, aiming to automate the extraction of key entities and red flags for contract review.

1.5. Outline

The remainder of this thesis is structured as follows: Chapter 2 introduces key concepts, including Employment Contracts, Prompting in LLMs, and Automatic Prompt Optimization (APO), with a focus on Instruction Optimization and Exemplar Selection. Chapter 3 presents our APO algorithm for fairness classification in German employment contracts, detailing the dataset, task formulation, prompt structure, the optimization objective and the algorithm itself. Chapter 4 outlines the experimental setup and results, benchmarking APO-optimized prompts against a baseline. Chapter 5 builds on these findings by addressing identified shortcomings and introducing a redesigned prompt methodology. Chapter 6 contextualizes our results within the existing literature, discussing implications, limitations, and future research directions. Finally, Chapter 7 summarizes key findings and outlines avenues for future work.

2. Background & Prelimimaries

This chapter presents the theoretical background for understanding the remainder of this thesis. We introduce our understanding of employment contracts (Section 2.1) and then go on to give a detailed account of the current state of Automatic Prompt Optimization (APO). This commences with an exploration of the role and structure of prompts in guiding large language models (Section 2.2). The chapter then defines the APO objective and distinguishes its two primary optimization strategies: Instruction Optimization (IO) and Exemplar Selection (ES) (Section 2.3). Emphasis is placed on analyzing the components and anatomy of LLM-based APO algorithms (Section 2.4), which form the theoretical foundation for this thesis.

2.1. Employment Contracts

We adhere to the definition of employment contracts as outlined by Wardas and Matthes [41], who describe them as legally binding agreements between employers and employees that define the terms and conditions of employment. According to their definition, these contracts, typically drafted by the employer, specify job responsibilities, compensation, work hours, benefits, and conditions for termination or modification. Their primary function is to protect the rights and obligations of both parties, ensuring clarity and preventing disputes. However, as Wardas and Matthes [41] highlight, evolving legislation, standardized contract templates, and individual modifications can introduce legally void clauses—provisions that are unenforceable and carry no legal effect. Identifying and rectifying such clauses requires legal expertise, leading many employers to outsource contract reviews to external law firms, which increases costs and time expenditure.

2.2. Prompting in Large Language Models

2.2.1. Large Language Models

Large Language Models (LLMs) [1, 2] have demonstrated exceptional performance across diverse natural language processing (NLP) tasks [56], including translation, text generation, and question answering. The development of LLMs has progressed through

four waves: statistical language models (SLMs), neural language models (NLMs) [57, 58, 59], pre-trained language models (PLMs) [60, 61, 62], and modern LLMs. SLMs, like n-gram models, suffered from data sparsity, a limitation addressed by NLMs using word embeddings to capture semantic relationships. PLMs advanced this further by pre-training on vast text corpora to generate generalized language representations, enabling task-specific fine-tuning with minimal labeled data. Modern LLMs, such as PaLM [63], LLaMA [64], and GPT-4 [65], scale these architectures to hundreds of billions of parameters [66, 67], leveraging transformers to process sequences in parallel with self-attention mechanisms for enhanced context understanding. This scalability has driven their success in specialized domains like medicine [68, 69, 70], education [71, 72, 73], law [17, 74, 75] and finance [76, 77].

However, LLMs have notable limitations. They lack memory, leading to inconsistent responses and an inability to retain information across interactions. They can facilitate societal biases [78], are computationally intensive, and can "hallucinate" [79], producing plausible but incorrect outputs.

2.2.2. Prompt Structure

Prompts are the primary mechanism for interfacing with LLMs, defining the context and constraints under which the models operate . A prompt serves as a structured input that guides the model's behavior for a given task, ensuring it generates relevant and coherent outputs [80, 81, 82, 83]. A typical prompt for LLMs can be conceptualized as a sequence [25]:

$$P(x) = [I, e_1, \ldots, e_k, c, x]$$

where I denotes the instruction that defines the task, $\{e_1, \ldots, e_k\}$ are k exemplars or demonstrations, c is the attached context and x is the input query. Prompts therefore usually consist of the following components with the concrete terms differing between scientific contributions [23]:

- **Instruction** I**:** A textual description of the task to be performed, specifying the type of output expected from the model. The instruction establishes the overall behavior of the model for the given task.
- **Exemplars** $\{e_1, \ldots, e_k\}$**:** Concrete input-output pairs (demonstrations) provided within the prompt. These exemplars are particularly important for In-Context Learning (ICL), where the model leverages the patterns shown in the examples to predict outputs for new, unseen inputs. In few-shot settings, exemplars contextualize the task by showing how input data maps to expected outputs, while in zero-shot scenarios, they are omitted, relying solely on the instruction.

- **Context** c**:** Additional background information provided to the model to improve its understanding of the task or input. This might include relevant definitions, constraints, or supplementary data.
- **Input query** x**:** The actual task-specific query that the model is expected to process, such as a sentence for translation, a document for summarization, or a clause for classification.

Figure 2.1 illustrates a typical template of how these components are put together for constructing a prompt for a grammatical task.

```
Construct grammatically correct sentences by following these rules:
1. A sentence must have a subject (noun) and a predicate (verb).
2. If the subject is singular, the verb must also be singular; if the subject is plural, the verb must
   also be plural.
3. Use proper punctuation: end a sentence with a period, question mark, or exclamation point
   depending on the context.

"The cat (noun) is sleeping (present continuous verb) on the mat."
Subject: "The cat" (singular)
Verb: "is sleeping" (matches singular subject)
Punctuation: Period at the end
<answer>The cat is sleeping on the mat.</answer>

(The boy / singular subject) + (is playing / verb phrase) + (in the park)

Subject: "The boy" (singular)
Verb: "is playing" (matches singular subject)
Punctuation: Period at the end
<answer>The boy is playing in the park.</answer>
```

Figure 2.1.: This figure illustrates a structured prompt example. The instruction I specifies the task to be performed, while the provided demonstration $\{e_1, \ldots, e_k\}$ exemplifies the application of the task rules. Both elements precede the input query x, enabling the model to generate an appropriate response in accordance with the outlined task guidelines.

2.2.3. Prompt Engineering

The interaction between users and large language models (LLMs) relies fundamentally on prompts [80]. Despite their simplicity, LLMs have shown to be highly sensitive to prompt format: Even minor variations in phrasing or structure can result in significant differences in model performance, often yielding unexpected or suboptimal outcomes

[84]. Furthermore, the effectiveness of a prompt is frequently model-dependent and task-specific, adding another layer of complexity to their design.

This inherent sensitivity makes the craft of designing effective prompts—known as prompt engineering—a crucial and increasingly recognized skill [85]. Prompt engineering involves strategically constructing and refining prompts to maximize model performance on a specific task [81]. Typically, this process involves a trial-and-error approach, where iterative adjustments are informed by an understanding of both the task at hand and the model's underlying behavior. While effective, this manual process is time-consuming and demands substantial expertise [24].

Over time, researchers have introduced systematic methodologies to improve prompt engineering practices. Techniques such as Chain-of-Thought Prompting [86], Self-Consistency [87], and Tree of Thoughts [88, 89] have been proposed to handle more complex reasoning tasks or improve reliability in outputs. Readers interested in exploring these methods further can refer to recent surveys [23, 90, 91] on advanced prompting strategies.

To address the limitations of manual prompt design, automatic prompt optimization (APO) has emerged as a promising alternative [38]. This approach leverages algorithmic techniques to iteratively refine prompts based on performance feedback. By automating the search for effective prompt structures, automatic optimization reduces reliance on human intervention and systematically uncovers prompts that might outperform manually designed ones.

2.3. Automatic Prompt Optimization

Automatic Prompt Optimization (APO) is a methodological approach aimed at systematically enhancing prompt performance by leveraging optimization techniques rather than relying solely on human expertise or trial-and-error methods [38]. In the context of Large Language Models (LLMs), APO seeks to determine the optimal prompt $P(x)$ that maximizes performance for a given task T, where the task is defined by a dataset $D = (X, Y)$ of input-output pairs. The performance of a prompt is typically evaluated using a metric g, such as accuracy or F1-score, on a validation dataset D_{val}.

Formally, APO can be defined as the process of finding the prompt $P^*(x)$ that maximizes the empirical performance on D_{val}, as given by the following optimization problem defined by Wan et al. [25]:

$$P^*(x) = \arg\max_{P(\cdot)\sim\mathcal{P}} \mathbb{E}_{(x_i,y_i)\sim D_{\text{val}}} \left[g(f_{\text{LLM}}(P(x_i)), y_i) \right]$$

where f_{LLM} represents the output of the LLM given the input x_i and the prompt P, and

$\mathcal{P}$ denotes the search space of possible prompts.

Optimization Dimensions

APO typically focuses on two primary dimensions of optimization [25]: instructions $\mathcal{I}$ and examples (or exemplars) $\mathcal{E}^k$. Instructions define the task-specific directives for the model, while examples serve as input-output demonstrations to guide the model's behavior. These two dimensions can be optimized either independently or jointly.

Instruction Optimization (IO): This involves refining the textual task description part $\mathcal{I}$ of a prompt to ensure clarity and task alignment. The search space in this case can be denoted as $P = \mathcal{I}$.

Exemplar Selection (ES): This focuses on selecting the most representative input-output pairs $\{e_1, \ldots, e_k\}$ to include in the prompt. Exemplars are crucial in few-shot learning scenarios and guide the model behavior. In this case the search space is defined as $P = \mathcal{E}^k$.

Combination of IO and ES: Approaches that tackle both IO and ES simultaneously have recently emerged [25, 37, 92, 93, 94]. In this case the prompt seach space is defined as $P = \mathcal{I} \times \mathcal{E}^k$. This joint objective results in a complex and high-dimensional combinatorial problem making it susceptible to local optima.

2.3.1. Instruction Optimization (IO)

Instruction Optimization (IO) methods can be categorized based on multiple criteria, including the nature of the optimization space, the accessibility of model gradients, and the availability of internal model parameters:

Optimization Space: Discrete vs. Continuous

Discrete optimization focuses on directly modifying token-level prompts, such as selecting, rearranging, or replacing words or phrases. These methods operate in the space of textual tokens, where each change is discrete and interpretable.
Continuous optimization operates in the latent embedding space of the model, adjusting continuous vector representations of prompts. This approach allows fine-grained control but often requires decoding the optimized embeddings back into human-readable text.

Model Access: Black-Box vs. White-Box Methods

Black-box methods treat the language model as a closed system, relying solely on observable outputs, such as log probabilities or performance metrics, to guide opti-

mization. These methods do not require access to internal parameters or gradients.
White-box methods leverage internal knowledge of the model, including its gradients and parameter structure, to directly optimize prompts. These methods are typically more computationally efficient but require greater access to the model.

Optimization Technique: Gradient-Free vs. Gradient-Based

Gradient-free approaches rely on heuristic-based search techniques, evolutionary algorithms, or other methods that do not depend on gradients. These approaches are particularly suitable for black-box settings where gradients are inaccessible.
Gradient-based methods use the gradients of the model's objective function to iteratively refine prompts. These approaches are predominantly applied in white-box settings, leveraging the mathematical structure of the model for precise optimization.

IO Paradigms and their Characteristics

Table 2.1 presents various instruction optimization (IO) paradigms, categorized by their optimization space, model access requirements, and optimization techniques.

Paradigm	Optimization Space	Model Access	Optimization Technique
Soft Prompt Tuning	Continuous	Both	Gradient-Based
Reinforcement Learning	Discrete	Both	Gradient-Based
Bayesian Optimization	Both	Black-Box	Both
LLM-based Optimizers	Discrete	Black-Box	Gradient-Free

Table 2.1.: Comparison of Instruction Optimization (IO) paradigms

Soft Prompt Tuning is a method that optimizes prompts in the embedding space, adjusting continuous vector representations while keeping the underlying model parameters frozen. Continuous black-box methods optimize prompts without accessing model gradients, relying on derivative-free techniques and external feedback. For example, subspace optimization reduces the complexity of the optimization process by operating in a low-dimensional subspace instead of the full embedding space [26]. This approach is further improved by projection-based optimization, which replaces uniform distributions with normal distributions for better performance [27]. In contrast,

continuous white-box methods utilize gradient information to directly optimize prompt embeddings. Prefix-tuning optimizes embeddings prepended to input text while keeping model parameters fixed, providing an efficient solution for task-specific adaptations [90, 95]. OptiPrompt extends this framework by introducing additional constraints that enhance optimization efficiency [29]. Furthermore, gradient-based approaches refine hard text prompts by bridging the gap between textual and embedding optimization, enabling precise adjustments to prompt representations [30].

Reinforcement Learning (RL) has been used to optimize discrete prompts by defining a policy network, typically a pretrained LLM, that generates candidate prompts evaluated based on task performance (e.g., classification accuracy). The reward signal guides the optimization process, similar to reinforcement learning from human feedback (RLHF) [31]. RLPrompt [32] uses this approach with a frozen LLM and a trainable feed-forward layer, generating prompts that are assessed by a downstream LLM. Despite non-differentiable reward signals, RLPrompt employs stabilization techniques to optimize prompts that often outperform traditional finetuning and soft prompt tuning, though the resulting prompts can be ungrammatical. Building on this, TEMPERA [94] introduces instance-dependent prompt optimization, dynamically adapting prompts at inference time through discrete edits (e.g., modifying instructions, adjusting exemplars, or altering verbalizers). Using RL, TEMPERA trains a policy to maximize performance improvement for each edit, demonstrating data efficiency and effectiveness in handling complex tasks.

Bayesian Optimization (BO) has emerged as a highly sample-efficient method for black-box optimization, gaining traction in prompt optimization for LLMs. One notable approach is InstructZero [33], which introduces an intermediate LLM between the soft prompt and the target LLM API. This auxiliary, open-source LLM converts soft prompts into human-readable instructions using additional task-specific information. These instructions are then evaluated by the target LLM, with their performance scores feeding into a BO framework that iteratively generates optimized soft prompts. Expanding on this concept, INSTINCT [76] replaces traditional Gaussian Processes in BO with neural networks, enabling more flexible and scalable optimization. Similarly, multi-armed bandit (MAB) approaches have been adapted for prompt optimization, as seen in [34], where an LLM-based strategy dynamically balances exploration and exploitation. Another recent development [35] integrates BO with LLMs by framing optimization tasks in natural language, allowing LLMs to iteratively propose refined solutions based on historical feedback.

LLM-based optimizers leverage large language models to handle the entire prompt optimization process, from initialization to iterative candidate refinement. This approach has gained prominence due to the increasing prevalence of black-box API-level access to LLMs, which limits direct access to internal parameters or gradients [24].

Conceptually simple, easy to implement, and highly effective, LLM-based optimization has emerged as a practical and popular paradigm. Given its accessibility and demonstrated success, this thesis focuses on LLM-based APO algorithms, providing a detailed analysis of their structure in Section 2.4.

2.3.2. Exemplar Selection (ES)

Exemplar selection (ES) is concerned with identifying the most relevant exemplars (input-output pairs) to construct effective few-shot prompts for in-context learning systems. Wan et al. [25] has highlighted the importance of ES, often surpassing instruction optimization (IO) in its impact on model performance. It has been argued that ES should accompany IO in most cases and deserves equal focus in APO.

Several paradigms exist for exemplar selection. **Similarity-Based Selection** [96, 97, 98] focuses on retrieving examples closest to the input using similarity metrics (e.g., cosine similarity in embedding space) to enhance contextual relevance. **Order Optimization** [99, 100, 101] investigates how the arrangement of examples impacts LLM performance, leveraging the sensitivity of outputs to example sequencing. **Influence Analysis** [46, 102] identifies the most impactful exemplars by measuring their contribution to task success, often through sensitivity analysis (e.g., output changes due to exemplar removal). **Model-Learned Selection** [103, 104] employs machine learning techniques to dynamically identify or generate optimal examples tailored to specific queries by leveraging reinforcement or supervised learning. **LLM-Generated Exemplars** [101, 105, 106, 107] rely on the LLM itself to produce examples, instead of using pre-labeled data, which are then refined or selected based on task performance. Finally, **Active Selection** [108] uses active learning principles to select examples that maximize task performance, systematically selecting examples that maximize information gain for the LLM.

2.4. The Anatomy of an LLM-based APO Algorithm

The development of LLM-based Automated Prompt Optimization (APO) algorithms can be traced back to Automatic Prompt Engineering (APE) [163], which introduced a self-referential framework for generating, scoring, and refining prompts. APE's recursive methodology laid the foundation for modern APO workflows by demonstrating the effectiveness of iterative prompt optimization. Building on this foundation, our framework illustrated in Figure 2.2 formalizes the LLM-based APO process into four distinct stages: Initialization (generating initial prompts), Expansion (creating variations to explore the prompt space), Selection (evaluating and selecting the most effective prompts), and Termination (identifying the optimal prompt and concluding

the process). These stages are modular and can be approached in isolation, enabling practitioners to customize APO algorithms by combining strategies across phases in a plug-and-play manner to meet specific requirements.

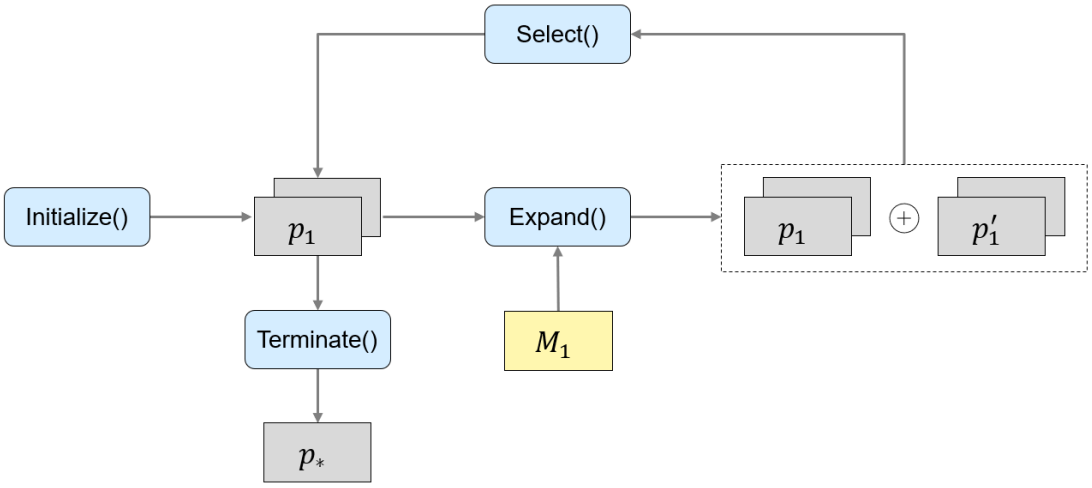


Figure 2.2.: Framework for LLM-based APO, comprising Initialization, Expansion (using meta-prompt M_1), Selection, and Termination to iteratively optimize prompts.

2.4.1. Step 1: Initialization

Effective initialization of instructions in automated prompt optimization (APO) algorithms plays a crucial role in ensuring robust performance and avoiding convergence to local optima. Three key paradigms have emerged for initializing instructions as a starting point for optimization:

- **Human-Generated Instructions:** The most traditional and widely used approach, where task-specific instructions are crafted based on human expertise and domain knowledge [38].
- **LLM-Generated Instructions:** Leveraging the capabilities of large language models, instructions are derived by reverse-engineering input-output pairs for the given task. Inspired by algorithms like Instruction Induction [109], this approach automates initialization which is especially useful if no prior human-written prompts are available.

- **Hybrid Strategies:** Combining the strengths of human and LLM-generated instructions, hybrid methods create a diverse initial instruction pool. For example, paraphrasing human-written instructions with an LLM introduces variation and improves the search space for optimization [110].

2.4.2. Step 2: Expansion

The expansion step aims to generate a set of new prompts, denoted as P_t, by building on the prompts from previous iterations, $P_{t-1}, \ldots, P_1$. The goal is to explore the prompt search space more broadly and steer it toward potential global optima. In the context of LLM-based Automatic Prompt Optimization (APO) algorithms, this process is implemented through a mutation operation $\mathcal{O}$ applied to each prompt selected for expansion in the current iteration. This transformation is facilitated by an Optimizer LLM, which operates on a classification prompt through a meta-prompt specifically designed for the mutation task. Table **??** summarizes the most prominent mutation operators, which can be categorized into four main classes. Each class is characterized by distinct meta-prompt designs tailored to the specific mutation strategy employed. Simplified meta-prompts for the four classes of operators are shown in Figure 2.3.

Category	**Papers**
Resampling-based Operators	[24][111][110]
Reflection-based Operators	[38] [39] [112] [113] [114][110][115][116][117][118][102]
Evolutionary Operators	[36][37][110][119][120]
Black-box Optimization Operators	[40]

Table 2.2.: Classification of most important mutation operators

Resampling-based Operators

Resampling-based operators as those employed by APE [24], PhaseEVO [110] and GPO [111] instruct the LLM to generate semantically similar prompt variations, making them

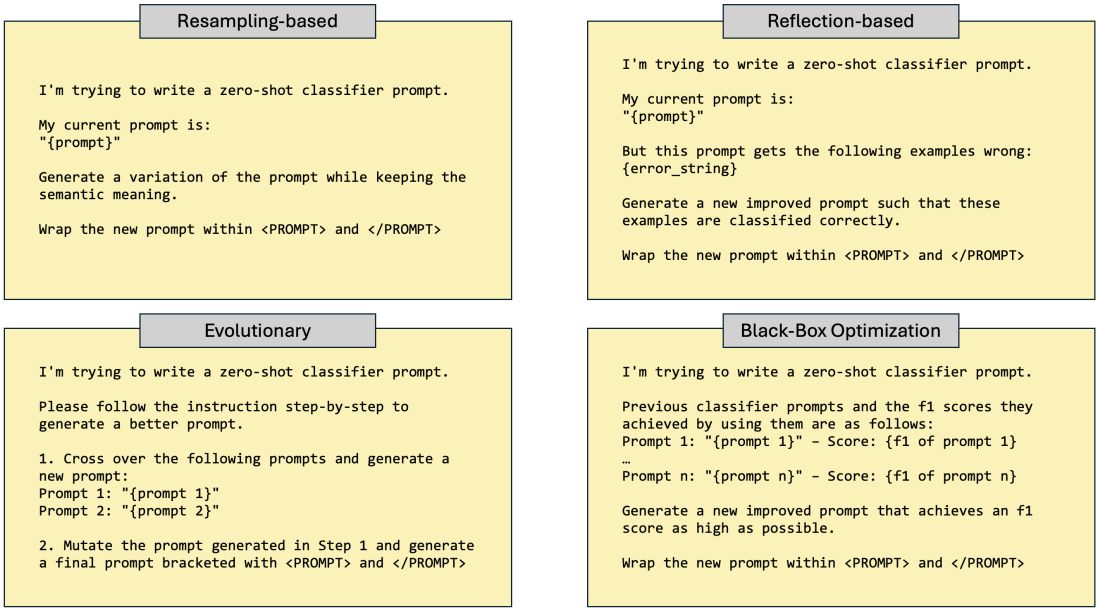


Figure 2.3.: Simplified meta-prompts for the four classes of mutation operators

particularly effective for local exploitation around high-performing candidates. The meta-prompt instructs the LLM to create a new prompt p_t that preserve the semantic meaning of the original prompt p_{t-1} while introducing diverse rephrasings, thereby remaining consistent with the original task intent:

$$p_t = \mathcal{O}(p_{t-1}),$$

where $\mathcal{O}$ is the function representing the transformation of the current prompt facilitated by the meta-prompt. By focusing on controlled, narrow semantic changes, resampling-based operators are prone for getting stuck in local optima due to their directionless updates.

Reflection-Based Operators

Reflection-based operators aim to address shortcomings in the current version of a prompt, denoted as p_{t-1}, by utilizing a large language model (LLM) to "reflect" on its own behavior. This process involves critiquing and correcting the prompts that the LLM uses for classifying instances. While primarily designed for downstream classification tasks, adaptations have been developed for non-classification tasks as well [121].

The process typically begins with a batch of instances that are classified using the

current prompt p_{t-1}. Inevitably, some of these instances will be misclassified, forming a set of errors E_{t-1}. These errors serve as the foundation for improving the prompt.

One method for reflection-based improvement involves directly instructing an Improver LLM with a meta-prompt. This meta-prompt includes the current version of the prompt and the set of errors E_{t-1}. The Improver LLM generates a new prompt p_t that incorporates refined instructions to address the misclassified inputs, formalized as:

$$p_t = \mathcal{O}(p_{t-1}, E_{t-1}),$$

where $\mathcal{O}$ again is the transformation function.

An alternative approach, pioneered by Pryzant et al. [38], introduces an intermediary step involving an Explainer LLM. In this method, the Explainer LLM is prompted to generalize the common characteristics of the misclassified inputs E_{t-1}, creating a localized improvement signal or guidance denoted as c_{t-1}:

$$c_{t-1} = \delta(p_{t-1}, E_{t-1}).$$

This guidance, referred to as "textual gradients," provides a semantic direction for updating the current prompt. The textual gradients, along with the current prompt p_{t-1} and an instruction, are then fed into an Improver LLM through a second meta-prompt. The Improver LLM uses this information to generate a new, improved prompt:

$$p_t = \mathcal{O}(p_{t-1}, c_{t-1}).$$

The practice of explicit reflection and refinement closely mirrors the trial-and-error nature of human-driven prompt engineering processes. However, reflection-based operators primarily aid in local optimization, similar to resampling-based operators. They can accelerate the convergence of a prompt optimization algorithm toward a local optimum but are less effective for global optimization.

Recent studies have highlighted limitations of reflection-based operators, particularly their difficulty in accurately identifying the root causes of errors. This limitation stems from the LLM's tendency to be influenced by its own prior knowledge rather than genuinely reflecting on the errors [122].

Evolutionary Operators

Evolutionary operators leverage principles from evolutionary algorithms, such as genetic algorithms and differential evolution, to generate new prompts by combining and mutating existing ones. These operators take two or more prompts as input and produce a new prompt through various recombination strategies.

A notable example of this approach is the **Crossover operator** [36], where two parent prompts, p_1 and p_2, are selected. The Optimizer LLM, guided by a meta-prompt, generates a new prompt p_t by crossing over components of the respective parent prompts. Formally, this can be expressed as $p_t = \mathcal{O}(p_1, p_2)$, where $\mathcal{O}$ denotes the crossover operation applied to the input prompts.

Another example is the **Differential Evolution operator** [36], which selects three prompts, p_1, p_2, and p_3, and generates a new prompt p_t by employing a four-step process. First, the corresponding parts in p_1 and p_2 with the same semantic meaning are identified. Second, these parts randomly mutated to create a new intermediate prompt. Third, the mutated prompt from the second step is combined with the third prompt, p_3, which is often the best-performing prompt in the current population. Lastly, a final crossover between the result from the third step and a prompt used for initialization is performed. The meta-prompt instructs the Optimizer LLM to execute these steps iteratively, resulting in a new prompt defined as $p_t = \mathcal{O}(p_1, p_2, p_3)$.

An advanced example of evolutionary prompt optimization is demonstrated by **Promptbreeder** [37], which extends the optimization process beyond the task-specific instruction prompt. Promptbreeder introduces a self-referential mechanism where not only the task prompt is optimized but also the mutation-prompt M responsible for improving task prompts. This is achieved through a higher-level hyper-mutation prompt H, which uses evolutionary operators to optimize M.

Black-Box Optimization Operators

Black-box optimization operators, such as OPRO [40], draw inspiration from the use of large language models (LLMs) as generic black-box optimizers for various optimization problems by defining an optimization problem in natural language and leveraging an LLM to iteratively propose new solutions. Each proposed solution is based on the history of prior solutions and their corresponding performance evaluations. Given that Automatic Prompt Optimization (APO) is fundamentally an optimization problem, these principles can be directly applied to optimize prompts.

In contrast to reflection-based operators, which refine the current prompt p_{t-1} by addressing specific misclassified instances or textual gradients, black-box optimization operators approach the problem holistically. They aim to generate a new prompt p_t by incorporating the entire optimization trajectory—consisting of prior prompts and their respective performance metrics—into the meta-prompt. This trajectory, combined with an instruction to improve the prompt, allows the LLM to generate a new prompt that maximizes the desired performance metric, such as F1-score in classification tasks.

At its core, the meta-prompt for black-box optimization operators consists of two primary components:

1. **Optimization Problem:** A natural language description of the optimization task. This includes details of the downstream LLM task, an explanation of how to interpret the optimization trajectory, and explicit instructions to generate a new prompt that maximizes the performance score for the downstream task.

2. **Optimization Trajectory:** A record of previous prompts and their corresponding objective values, i.e., the performance metric achieved by each prompt when applied to the downstream task.

The transformation function $\mathcal{O}$ achieved through this meta-prompt is described as:

$$p_t = \mathcal{O}(p_{t-1}, \ldots, p_1),$$

where the choice of which previous prompts to include in the optimization trajectory is an algorithmic design decision.

2.4.3. Step 3: Selection

The selection step is critical in identifying and maintaining the most promising prompt candidates during each iteration of the APO process. These selected prompts are subsequently utilized in the `expand()` function to generate new offspring candidates. Selection strategies can broadly be categorized into two paradigms: distance-based and search-based approaches.

Search-based Selection

Search-based selection strategies leverage established search algorithms to navigate the large and complex prompt search space effectively. These methods aim to balance exploration (identifying diverse prompt candidates) and exploitation (refining high-performing candidates). Two prominent strategies in the APO literature are:

- **Beam Search [24, 38, 39]:** This approach maintains a fixed number of top-performing prompt candidates, referred to as the beam width k, in each iteration. Candidates are selected based on their scores computed via a scoring function g, such as accuracy, F1-score, or other task-specific metrics. The top k candidates are retained for subsequent expansion in the `expand()` function.

- **Monte Carlo Tree Search (MCTS) [39]:** In MCTS, a search tree is built where each node represents a prompt candidate, and edges correspond to transitions between candidates achieved via expansion operators. A child node (prompt candidate) is selected for expansion based on a balance metric such as the Upper Confidence

Bound (UCB), which weighs exploitation (high-performing nodes) against exploration (less-visited nodes). The selected node is expanded by applying one of the expansion operators to generate a new prompt candidate. A simulated performance score like F1-Score for the new prompt candidate is obtained by applying it to the downstream task. Finally, the simulated score is backpropagated to update the values of ancestor nodes in the tree.

Distance-based

Prompts for expansion are selected based on a distance metric $d(p_i, p_j)$, like Cosine Similarity [36] or Hamming Distance [110], to measure similiarity or dissimilarity between prompt candidates. This selection strategy is exclusively used for selecting parent prompts in case of evolutionary operators as they require at least two parents:

- **Closeness:** Select $p_i, p_j \in P$ such that $d(d_i, d_j) < t$, where t is a threshold, or choose the k closest prompts. This strategy is used to maintain the distribution of the current prompt population.
- **Distinctness:** Select $p_i, p_j \in P$ such that $d(d_i, d_j) > t$, where t is a threshold, or choose the k most distinct prompts. This strategy is used to combine ascpects of distinct prompt to explore a broader search space.

2.4.4. Step 4: Termination

The termination step marks the conclusion of the APO algorithm, wherein the optimal prompt p^* is selected based on the accumulated results of the iterative `expand()` and `select()` operations. The literature outlines two primary approaches for terminating the APO process: **Scheduled Termination** relies on a predefined computational budget, typically specified as a fixed number of T iterations. After T iterations of the `expand()` and `select()` steps, the algorithm ceases further optimization, and the best-performing prompt p^* is selected based on the scoring function g, as determined by the employed search strategy. Scheduled termination may risk halting the optimization prematurely if the optimal prompt has not yet been discovered. Unlike scheduled termination, **Early Stopping** introduces a dynamic criterion that terminates the algorithm when performance improvements stagnate. Specifically, the algorithm stops if, over a predetermined number of consecutive iterations, there is no performance gain or if the observed improvement falls below a predefined threshold. Performance gain can be evaluated either as an average improvement across a subset of prompt candidates or as the improvement in the best-performing prompt's score. Early stopping balances computational efficiency with optimization efficacy by preventing unnecessary iterations once diminishing returns are observed.

3. Proposed Framework for Automatic Prompt Optimization

This chapter presents the design of an Automatic Prompt Optimization (APO) approach aimed at enhacing the classification of clauses within German employment contracts. The objective of this method is to iteratively refine prompts, composed of components and policies, to achieve optimal performance in assessing the legality of clauses. The chapter is organized into several sections: we begin by formally defining the problem (Section 3.1.2 and the dataset (Section 3.1.1) used. We then introduce the structure of the prompt (Section 3.1.3), followed by a detailed explanation of the APO algorithm (Section 3.2), including its initialization, iterative improvement, and selection phases.

3.1. Problem Formulation

3.1.1. Dataset

The dataset used for our APO algorithm comprises a clause corpus obtained from German employment contracts [41] and associated legal policies.

Clauses The clause corpus C contains 893 clauses, each assigned one of three labels: *fair*, *unfair*, or *void*. The clauses were extracted from contracts originating from a law firm's client data. The labels were determined by certified lawyers from a small-size German law firm specialized in Economic Law in 3 annotation rounds. Each clause is further categorized into one of 14 categories. On top of the category, each clause comes with the title of the section in the contract where it was located and in case of a negative label (unfair or void) and explanation for this label. For a more detailed explanation of the labels the reader my refer to [41].

Policies In addition to the clauses, the dataset includes a set of 20 policies, denoted as T. These policies are formulated as legal fuzzy norms, consisting of a legal condition (Tatbestand) and a legal consequence (Rechtsfolge). Policies are designed to assist in identifying void clauses. For instance, a policy might specify that "any clause in general terms and conditions that establishes a general prohibition on garnishment

or assignment of employee claims is void according to § 308 Nr. 9 lit. a BGB." It is important to emphasize that the policies are not exhaustive legal norms but are crafted with the primary purpose of reliably identifying void clauses within the dataset. The policies were developed and provided by the same law firm that provided the clause corpus. They were then assigned to one or more clause categories by us.

Category	**# Clauses**	**# Policies**	**Label Distribution (%)**
Termination	92	0	87 / 9 / 4
Penalty Clause	117	5	70 / 12 / 18
Statute of Limitations	30	0	53 / 7 / 40
Compensation	127	4	67 / 6 / 27
Format	24	1	46 / 29 / 25
Cut-off Periods	9	5	0 / 11 / 89
Illness	70	1	81 / 7 / 12
Garnishment	20	2	40 / 5 / 55
Execution	108	0	89 / 10 / 0
Tasks	87	0	91 / 3 / 6
Vacation	59	0	78 / 17 / 5
Overtime	8	2	75 / 12.5 / 12.5
Other	142	2	92 / 7 / 1

Table 3.1.: Distribution of clause categories, policies, and labels. Label distribution is shown as percentages of Fair / Unfair / Void clauses.

3.1.2. Task

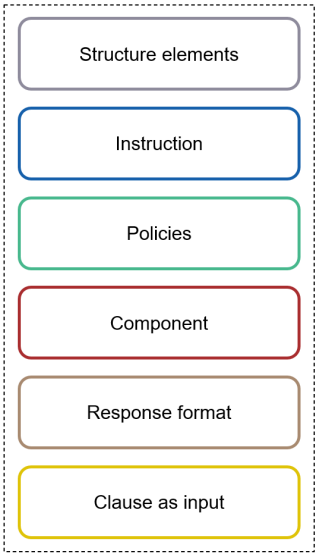
The multi-label classification task involves the use of a Large Language Model (LLM) to evaluate clauses $(x, y) \in \mathcal{C}$ regarding their fairness and legal permissibility. Specifically, given a prompt $P(x)$ generated according to the framework outlined in Section 3.1.3, the LLM processes $P(x)$ to produce an output $f_{\text{LLM}}(P(x))$, representing the predicted label for the clause. Each clause is assigned one of three labels: *fair*, *unfair*, or *void*, with the true label denoted by $y \in \{\text{fair}, \text{unfair}, \text{void}\}$. The classification performance is evaluated across the entire dataset $\mathcal{C}$ using various performance metrics $g(f_{\text{LLM}}(P(x)), y)$. These metrics quantify the agreement between the predicted labels $f_{\text{LLM}}(P(x))$ and the ground truth y. Standard performance metrics for classification tasks, such as accuracy, precision, recall, and F1-score can be used in this context.

3.1.3. Prompt Structure

We classify clauses with an LLM using prompts as input. Denoting a given clause x and its label y, a prompt $P(x)$ may be represented as:

$$P(x) = [s, t_1, \ldots, t_k, x] \tag{3.1}$$

where s denotes a *component* and $t_1, \ldots, t_k$ denote k *policies*. The main function of the *component* is to explain the characteristics of a fair or unfair clause whereas the function of the *policies* is to assist in identifying void clauses. We show a template organizing these components in Figure 3.1. Note that the prompt template has building blocks which are fixed and therefore not considered in the equation. In the following $C = S \times T'$ where $T' \subset T$ is referred to as the set of possible candidates.




```
<anweisung>
Du bist ein deutscher Anwalt, spezialisiert auf Wirtschaftsrecht, und bewertest Klauseln der Kategorie {category} aus
Arbeitsverträgen hinsichtlich ihrer rechtlichen Zulässigkeit und Fairness.

Arbeite strikt Schritt für Schritt und beachte die Regeln, die dir zusammen mit der Klausel zur Verfügung gestellt werden.
Diese Regeln wurden von professionellen Anwälten erstellt und sind immer als korrekt anzusehen.

<vorgehen>
1. Prüfe im ersten Schritt, ob der Tatbestand von einer oder mehrerer der Regeln zutrifft. Falls ja, wende die Regeln an und
wäge sie ab, um die entsprechende Klassifikation (‚unbedenklich', ‚potentiell unzulässig' oder ‚unzulässig') der Klausel zu
bestimmen.

2. Falls keine der Regeln zutrifft oder keine Regeln existieren, wende die bereitgestellten "Allgemeinen Prinzipien" zur
Klassifikation an. In diesem Fall sollst du die Klausel nur entweder als 'unbedenklich' oder 'potentiell unzulässig' nicht aber
als 'unzulässig' klassifizieren.
</vorgehen>

<regeln>
{policies}
</regeln>

<Allgemeine Prinzipien>
{component}
</Allgemeine Prinzipien>

Befolge bei deiner Antwort bitte das gegebene Antwortformat und gib an, welche Regeln du angewendet hast, um die Klausel zu
klassifizieren.
</anweisung>

<antwortformat>
      <bewertung>Gib hier die abschließende Klassifikation ein: {classifications}.</bewertung>
      <regeln>Gib hier durch Kommas getrennt ausschließtlich die IDs derjenigen Regeln an, die du angewendet hast, um die
              Klausel zu klassifizieren. Wenn keine der Regeln Anwendung findet, also die allgemeinen Prinzipien angewendet
              wurden, dann bleibt die Liste leer
      </regeln>
      <erklaerung>Gib hier eine kurze Erklärung an, wie du zu deiner Bewertung gekommen bist</erklaerung>
</antwortformat>

<frage>Ist die folgende Klausel ‚unbedenklich', ‚potentiell unzulässig' oder ‚unzulässig'?<frage>

<klausel>{clause}</klausel>
```

Figure 3.1.: Structured prompt template guiding the LLM to classify employment contract clauses as 'fair', 'unfair', or 'void', using predefined instructions, policies, and components.

3.1.4. APO Objective

We extend the principles of Automatic Prompt Optimization (APO) by tailoring prompts to specific groups of clauses similar to [42]. Each cluster $\mathcal{C}_i$, representing a subset of clauses with similar characteristics, is assigned a cluster-specific prompt $P_i^*(x)$, ensuring the optimization focuses on cluster-specific nuances. For instance, clauses in a cluster

concerning sick leave policies ($\mathcal{C}_1$) may require prompts emphasizing the validity of medical certificates or deadlines for notification. In contrast, clauses in a cluster on overtime compensation ($\mathcal{C}_2$) may need prompts focusing on compliance with labor laws regarding fair compensation. The objective is formalized as:

$$P_i^*(x) = \arg \max_{P(\cdot) \sim \mathcal{P}} \mathbb{E}_{(x,y)\sim \mathcal{C}_i} \left[g(f_{\text{LLM}}(P(x)), y) \right] \tag{3.2}$$

Here, $P(\cdot)$ represents a candidate prompt from the search space $\mathcal{P}$, $f_{\text{LLM}}(P(x))$ denotes the LLM's output for the transformed input $P(x)$, and $g(\cdot, \cdot)$ is the performance metric comparing the LLM's prediction to the ground truth y as assigned by the lawyers. Clause-label pairs (x, y) are sampled from the specific cluster $\mathcal{C}_i$. The search space $\mathcal{P}$ spans combinations of components and policies and encompasses a discrete and intractable space of arbitrarily large dimension, making the optimization problem in Eq. 3.2 extremely difficult.

3.2. APO Algorithm

Our APO algorithm as shown in Alg. 1 aims to optimize prompts by iteratively refining candidates, each consisting of a component and an assigned set of policies, to maximize classification performance for the legality assessment of German employment contract clauses. The algorithm employs a beam search approach over the candidate space $S \times T$ (components S and policies T), optimizing at the cluster level, where clauses are partitioned into distinct categories or embedding-based clusters. The optimization process follows an iterative improvement paradigm, structured into three phases: clustering of clauses and policies, initialization of candidate populations, and iterative candidate improvement.

In **Phase 1: Clustering of Clauses and Policies**, we partition clauses into clusters either based on predefined categories or using embedding-based methods like k-means. The initial policies created by the lawyers are then assigned to these clusters.

Algorithm 1 APO Classification German Employment Contracts

Require: Set of clauses C, set of policies P, number of clusters n, number of candidates per cluster $n_{\text{population}}$, number of iterations T, score function S, batch size b

Partition clauses into disjoint clusters:
$\{C_1, C_2, \ldots, C_n\} \leftarrow \texttt{Cluster\_Clauses}(C)$
Assign each policy to one or more clusters:
Policy_Mapping $\leftarrow \texttt{Assign\_Policies\_To\_Clusters}(\{C_1, \ldots, C_n\}, P)$
for all clusters $i \in \{1, 2, \ldots, n\}$ **do**
    Divide clauses in C_i into batches of size b:
    batches$_i \leftarrow \{B_1, B_2, \ldots, B_k\}$, $k = \lceil |C_i|/b \rceil$
    Create initial candidates for cluster:
    candidates$_i^0 \leftarrow \texttt{Initialize_Candidates}(C_i, \text{Policy_Mapping}[C_i], n_{\text{population}})$
    Evaluate scores for all initial candidates within cluster:
    $S_i^0 \leftarrow \{s_{k,i}^0 = F(c_{k,i}^0, C_i) \mid c_{k,i}^0 \in candidates_i^0\}$
end for
for iteration $j \in \{1, 2, \ldots, T\}$ **do**
    for all clusters $i \in \{1, 2, \ldots, n\}$ **do**
        for each batch $B_k \in$ batches$_i$ **do**
            $D \leftarrow candidates_i^{j,k-1}$ (candidates of previous batch)
            for all $c = (s, T') \in candidates_i^{j-1}$ **do**
                Update component based on clauses in in B_k:
                $s_{new} \leftarrow \texttt{Refine\_Component}(c, B_k)$
                Update policies based on clauses in in B_k:
                $T'_{new} \leftarrow \texttt{Refine\_Policies}(c, B_k)$
                Add new candidate to D:
                $D \leftarrow D \cup \{(s_{new}, T'_{new})\}$
            end for
            Crossover candidates with greatest Hamming distance:
            $D \leftarrow D \cup \texttt{Crossover}(D)$
            Select candidates with highest scores for next iteration:
            $candidates_i^{j,k} \leftarrow \texttt{Select}(D, n_{\text{population}})$
        end for
    end for
end for
return Final candidates with their respective evaluated scores $\{(candidates_1^T, S_1^T), \ldots, (candidates_n^T, S_n^T)\}$ for all clusters

In **Phase 2: Initialization of Candidate Populations**, we generate an initial popula-

tion of candidates for each cluster to ensure diversity. This initialization is achieved through two approaches:

- **Human-Crafted Standard Component**: Candidates are initialized with a predefined standard component s_0.
- **Reverse Engineering from Labeled Clauses**: Candidates are initialized with components derived from the labeled clauses in the cluster.

In **Phase 3: Iterative Candidate Improvement**, the candidates undergo continuous refinement through a sequence of operations:

- **Local Exploitation via Candidate Reflection**: Components and policies are updated based on batches of clauses within each cluster.
- **Global Exploration via Candidate Crossover**: Candidates with the greatest Hamming distance are combined to explore new component configurations.
- **Selection**: The best-performing candidates (based on a score function S) are selected to proceed to the next iteration.

Our algorithm integrates *exploration* and *exploitation* strategies to balance global search efficiency with local convergence speed. The *exploration* phase uses the evolutionary crossover operator to broadly explore the candidate space and avoid local optima. The *exploitation* phase uses feedback-driven refinement to incrementally improve candidates within localized regions of the search space. This dual approach facilitates an efficient search, enabling incremental performance improvements.

3.2.1. Phase 1: Clustering of Clauses and Policies

In this phase, clauses are grouped into clusters, and relevant policies are assigned to these clusters. The clustering process can be performed in one of two ways: (1) using predefined categories or (2) using K-Means clustering on clause embeddings. The choice of clustering method directly influences the policy assignment strategy. Additionally, clustering can be omitted entirely, resulting in a single cluster containing all clauses and policies.

Clustering of Clauses

The goal of clustering is to group similar clauses together to enable more targeted prompt optimization. We support two approaches for clustering clauses:

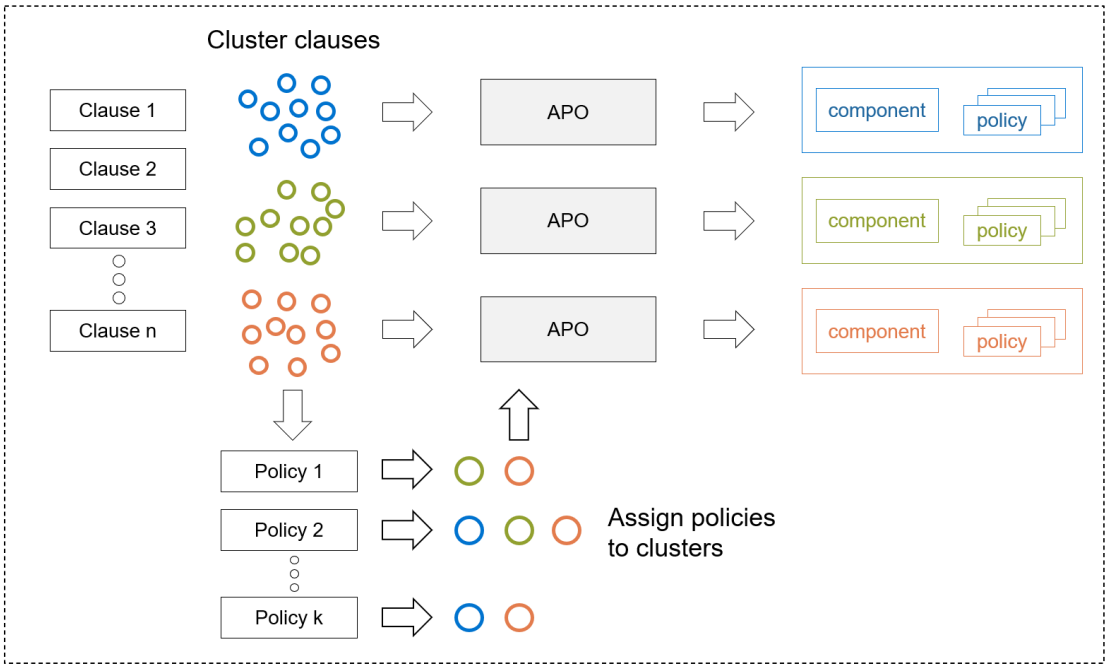


Figure 3.2.: Overview of the APO approach, illustrating the clustering of clauses, assignment of policies to clusters, and the subsequent optimization of components and policies for each cluster individually.

Category-Based Clustering: Clauses can be clustered based on predefined categories such as *Form*, *Krankheit*, *Durchführung*, etc. In this approach, each clause is assigned to a cluster corresponding to its category assigned by the lawyers as shown in 3.1. The clustering process is outlined in Alg. 2.

Algorithm 2 *Cluster_Clauses*$(\cdot)$ based on categories

Require: Set of clauses C

1: Initialize $\{C_1, C_2, \ldots, C_n\}$ as empty sets
2: **for** $c \in C$ **do**
3: Assign c to a cluster C_i based on predefined clause category (Form, Krankheit, Durchführung ...)
4: **end for**
5: **return** $\{C_1, C_2, \ldots, C_n\}$

Embedding-Based Clustering via K-Means: Alternatively, clauses can be clustered using K-Means in the embedding space. Clause embeddings are generated using an off-the shelf embedding model (e.g., *Sentence-BERT* [123] or *OpenAI's text-embedding-ada-002* [124]), and K-Means clustering is applied to partition the clauses into n clusters. The process is detailed in Alg. 3. The algorithm first computes the embeddings $E = \{\text{Embed}(c) \mid c \in C\}$ and then applies K-Means to these embeddings to produce n clusters.

Algorithm 3 *Cluster_Clauses*$(\cdot)$ based on K-Means clustering

Require: Set of clauses C, number of clusters n
Ensure: Disjoint clusters $\{C_1, C_2, \ldots, C_n\}$
1: Compute clause embeddings:

$$E = \{\text{Embed}(c) \mid c \in C\}, \quad \text{where } E \text{ is the set of clause embeddings}$$

2: Perform K-Means clustering on embeddings E with n clusters:

$$\{C_1, C_2, \ldots, C_n\} = \text{KMeans}(E, n)$$

3: **return** $\{C_1, C_2, \ldots, C_n\}$

No Clustering: If clustering is omitted, all clauses are grouped into a single cluster. In this case, we consider one "big" cluster containing all clauses in C. All initial policies are assigned to this single cluster, and we optimize a single classification prompt for the entire clause population.

Assignment of Policies to Clusters

The assignment of policies to clusters depends on the clustering approach chosen, as described in Alg. 4 and Alg. 5. In all cases, the algorithm initializes an empty mapping and assigns each policy T_j to at least one cluster.

Category-Based Clustering: When using category-based clustering, policies are manually mapped to categories based on predefined relevance. This mapping is defined by domain knowledge (as shown in Table 3.1) and ensures that each cluster C_i receives the appropriate set of policies T_i'. The process is outlined in Alg. 2.

Algorithm 4 *Assign_Policies_To_Clusters*$(\cdot)$ based on categories

Require: Clusters $\{C_1, C_2, \ldots, C_n\}$, set of policies $T = \{T_1, T_2, \ldots, T_k\}$, mapping of policies to categories
Ensure: Policy mapping: Policy_Mapping $= \{C_i \mapsto T_i' \mid T_i' \subseteq T\}$
Initialize Policy_Mapping $= \{C_i \mapsto \emptyset \mid i = 1, \ldots, n\}$
for $T_j \in T$ **do**
  Retrieve categories A_j associated with T_j
  for C_i such that C_i corresponds to a category in A_j **do**
    Add T_j to Policy_Mapping$[C_i]$: $T_i' \leftarrow T_i' \cup \{T_j\}$
  end for
end for
return Policy_Mapping

Embedding-Based Clustering: If K-Means clustering is used, policies are transformed into the embedding space and assigned based on proximity to cluster centroids. Specifically, each policy T_j is assigned to the three closest cluster centroids. This ensures that each policy is mapped to the clusters most relevant to it based on embedding similarity. The detailed steps are provided in Alg. 5.

Algorithm 5 *Assign_Policies_To_Clusters*$(\cdot)$ with K-Means Clustering

Require: Clusters $\{C_1, \ldots, C_n\}$, policies $T = \{T_1, \ldots, T_k\}$
Ensure: Policy mapping: Policy_Mapping $= \{C_i \mapsto T_i' \mid T_i' \subseteq T\}$
$E_T \leftarrow \{\text{Embed}(T_j) \mid T_j \in T\}$
Policy_Mapping $\leftarrow \{C_i \mapsto \emptyset \mid i = 1, \ldots, n\}$
for $T_j \in T$ **do**
  $A_j \leftarrow \text{NearestClusterCenters}(\text{Embed}(T_j), 3)$
  for $C_i \in A_j$ **do**
    Policy_Mapping$[C_i] \leftarrow$ Policy_Mapping$[C_i] \cup \{T_j\}$
  end for
end for
return Policy_Mapping

No Clustering: If clustering is omitted, all clauses are grouped into a single cluster containing the entire clause population. In this case, all policies $T = \{T_1, T_2, \ldots, T_k\}$ are assigned to this single cluster. This simplifies the policy mapping process, as the full set of policies is used for the single candidate population.

3.2.2. Phase 2: Initialization of Candidate Populations

Our objective in this phase is to generate a diverse set of candidates as the initial population for each cluster, ensuring a broad exploration of the joint space of components and policies. Each candidate consists of a component and a set of policies assigned to the cluster. We employ two distinct initialization strategies for the components: a predefined standard component and a dynamically generated component via reverse engineering. These strategies contribute equally, each initializing half of the candidates within the population for each cluster.

Algorithm 6 illustrates this process, ensuring a balance between human-crafted components and those derived through LLM-driven reverse engineering. Importantly, while the component varies depending on the initialization strategy, the policy set T_i' assigned to each candidate remains consistent within a cluster.

Algorithm 6 Initialize_Candidates(·) - line 5 in Algorithm 1

Require: Cluster C_i, policies assigned to the cluster T_i', number of candidates $n_{\text{population}}$, standard component s_0
Ensure: A list of candidates for C_i: candidates_i

```
Initialize candidates_i^0 ← ∅
for j ← 1 to n_population do
    if j ≤ n_population mod 2 then
        Assign standard component s_0 to the candidate:
        candidate ← (s_0, T_i')
    else
        Assign component from instruction_induction(C_i) to the candidate:
        candidate ← (instruction_induction(C_i), T_i')
    end if
    Add candidate to candidates_i^0:
    candidates_i^0 ← candidates_i^0 ∪ {candidate}
end for
return candidates_i^0
```

Initialization with Human-Crafted Standard Component

In this approach, candidates are initialized using a predefined, human-crafted standard component s_0. Each candidate c is initialized as:

$$c \leftarrow (s_0, T_i') \tag{3.3}$$

with s_0 being shown in figure 3.3

```
Du bewertest die Klausel als 'potentiell unzulässig' wenn sie den Arbeitgeber oder den Arbeitnehmer unangemessen
benachteiligt oder besonders ungewöhnliche Formulierungen enthält.
```

Figure 3.3.: Standard component s_0

Initialization by Reverse Engineering from Labeled Clauses

In this approach, candidates are initialized with a dynamically generated component tailored to the specific characteristics of the clauses within each cluster. This is achieved through the 'instruction_induction' function, which employs an LLM to generate a component by reverse-engineering from a set of clause-label pairs within the cluster. Given a set of clauses C_i, let S_i be a subset of labeled clauses:

$$S_i = \{(x_1, y_1), \ldots (x_m, y_m)\} \subseteq C_i \tag{3.4}$$

The 'instruction_induction' function uses an LLM $\mathcal{L}$ and a meta-prompt to generate a component informed by the labeled clauses and the label distribution within the cluster. The candidate c is then initialized as:

$$c \leftarrow (\texttt{instruction\_induction}(C_i), T_i') \tag{3.5}$$

The meta-prompt, which guides the LLM to produce a component, is shown in Fig. 3.4.

3.2.3. Phase 3: Iterative Candidate Improvement

This phase involves an iterative optimization process where, in each iteration, the current candidates in each cluster population are refined to generate new candidates. The refinement is performed using a batch of clauses sampled from the corresponding cluster. After generating new candidates, the most promising ones are selected for the next iteration. The iterative nature of this approach facilitates comprehensive exploration of the candidate search space while progressively improving the candidate populations.

3.3.1 Local Exploitation via Candidate Reflection

In this step, we generate new candidates by refining the component and policies of the existing candidates in the population. This process leverages the self-reflection

```
Du bist ein Prompt Engineer, spezialisiert darauf, Prompt-Formulierungen für einen LLM-Klassifizierer zu erstellen. Der
Klassifizierer ist darauf spezialisiert, Klauseln der Kategorie {category} aus Arbeitsverträgen hinsichtlich ihrer rechtlichen
Zulässigkeit und Fairness zu bewerten und sie in einer der folgenden Kategorien zu klassifizieren: „unbedenklich“, „potentiell
unzulässig“, „unzulässig“.

### Inputdaten
- **Klauseln**: Eine Teilmenge der Klauseln, die klassifiziert werden soll, befindet sich unter ### Klauseln zusammen mit der
tatsächlichen Bewertung durch Anwälte, die als „ground truth“ gilt (als „Bewertung“ aufgeführt).
- **Verteilung**: Tatsächliche Verteilung der Klauseln aus der Kategorie {category} in die Bewertungskategorien „unbedenklich“,
„potentiell unzulässig“, „unzulässig“. Die Verteilung ist als wahr anzunehmen, weil sie von professionellen Anwälten kommt.

### Deine Aufgabe:
Erstelle eine Version der „Klassifizierungskomponente“, die den Klassifizierer so anleitet, dass die angehängten Klauseln vom LLM-
Klassifizierer korrekt eingestuft werden. Die Komponente sollte:
1. Die spezifischen Bedingungen für die Klassifikationen „unbedenklich“ und „potentiell unzulässig“ so präzisieren, dass alle
Klauseln korrekt klassifiziert werden. Ein besonderes Augenmerk soll nicht auf den eindeutigen, sondern auf den Grenzfällen liegen.
2. Inkorporiere auch die Verteilung der Klauseln in die einzelnen Bewertungskategorien bei der Erstellung, allerdings nur indikativ,
ohne die konkrete Verteilung in Zahlen zu nennen.

Verfasse die neue Klassifizierungskomponente ohne Einleitungsformulierungen. Die Komponente soll direkt beginnen mit den Kriterien
und keine zusätzlichen Titel oder Erläuterungssätze wie „Die Klassifizierungskomponente wird wie folgt definiert“ enthalten.

Nutze das folgende Format, um die neue Komponente abzugrenzen:
<START>...Das ist ein Platzhalter für die neue Komponente...<ENDE>

### Klauseln:
{clauses}

### Verteilung:
{clause_distribution}

Erstelle die Komponente so, dass alle bereitgestellten Klauseln vom Klassifizierer richtig bewertet werden.
```

Figure 3.4.: Meta-prompt for guiding the LLM to derive an initial classification component. The placeholders {clauses} and {clause_distribution} are replaced with the set of clauses and their label distribution, respectively.

capabilities of large language models (LLMs). Specifically, we harness the LLM's ability to self-diagnose and correct its own prompt shortcomings by focusing on misclassified clauses.

Given a cluster C_i and a batch of clauses $B = \{b_1, b_2, \ldots, b_k\}$, the refinement process proceeds as follows:

1. Classification Using Current Candidates: Each candidate c in the population is applied to the batch B. This is done by inserting the candidate's component and policies into the classification prompt template (shown in Fig. 3.1) and using the resulting prompt to classify the clauses in B.

2. Error Collection: We identify the misclassified clauses for each candidate. Specifically, we collect the set of clauses where the LLM's classification, using the current candidate's prompt configuration, deviates from the ground-truth labels. This set of errors is denoted as:

$$E_c = \{(x_i, y_i) \mid x_i \in B, \text{LLM}(P_c(x_i)) \neq y_i\}$$

3. Meta-Prompt-Based Refinement: The collected errors are fed into two meta-prompts:

- Component Refinement Meta-Prompt δ_c (Fig. 3.6): Instructs the LLM to refine

the component of the candidate to address the identified errors of fair or unfair clauses.

- Policy Refinement Meta-Prompt δ_p (Fig. 3.7: Instructs the LLM to refine the policies of the candidate to address the identified errors of unfair clauses.

.

4. Refinement Execution: The LLM processes these meta-prompts and generates updated components and policies. The new candidate c' is then constructed with these refined elements.

This procedure ensures that each candidate undergoes targeted refinement based on specific classification errors. By iteratively applying this reflection process, the candidates are incrementally improved, converging toward higher-performing prompt configurations. The complete workflow is illustrated in Fig. 3.5.

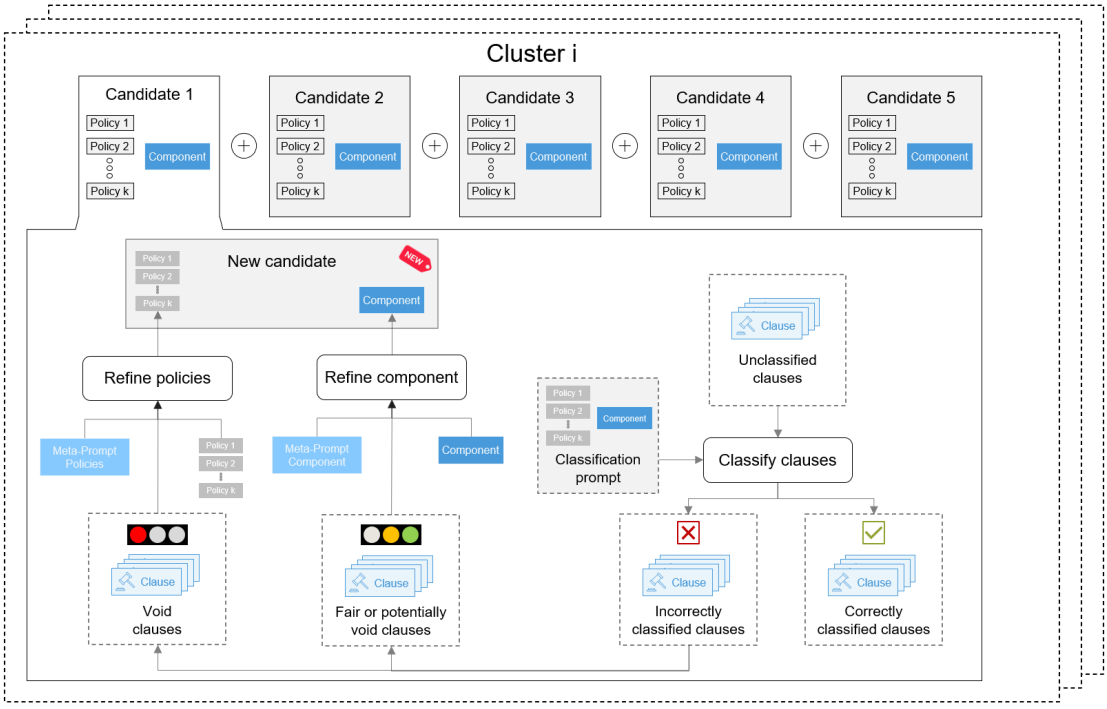


Figure 3.5.: Candidate refinement process, where misclassified clauses guide the iterative adjustment of policies and components using meta-prompts, generating improved candidates.

Component Refinement In the component refinement step, we update the component of each candidate based on the classification errors identified during the current iteration. The process specifically targets clauses that are labeled as fair (0) or unfair (1) and were misclassified by the LLM. Given a candidate $c = (s, T)$ with component s and policies T, and a batch of clauses $B = \{b_1, b_2, \ldots, b_k\}$, the refinement proceeds as follows:

1. Error Collection: During the classification process, errors are collected for all misclassified clauses in B. From this set of errors, we filter the clauses labeled as fair or unfair.

Refinement Using Meta-Prompt: The filtered errors are fed into the component refinement meta-prompt δ_c. This meta-prompt instructs the LLM to generate an improved component s_{new} that addresses the shortcomings identified by these specific errors. The refinement process is described in Alg. 7.

```
Du bist ein Prompt Engineer, spezialisiert darauf, Prompt-Formulierungen für einen LLM-Klassifizierer zu verbessern. Der Klassifizierer ist darauf
spezialisiert, Klauseln der Kategorie {category} aus Arbeitsverträgen hinsichtlich ihrer rechtlichen Zulässigkeit und Fairness zu bewerten und sie
in einer der folgenden Kategorien zu klassifizieren: „unbedenklich“, „potentiell unzulässig“ oder „unzulässig“.

In der letzten Iteration wurde der Klassifizierer-Prompt verwendet, um mehrere Klauseln zu klassifizieren. Einige dieser Klauseln wurden jedoch
falsch klassifiziert. Für jede falsch klassifizierte Klausel sind folgende Informationen angegeben:
- Die tatsächliche Bewertung durch Anwälte, die als „ground truth“ gilt (als „Bewertung“ aufgeführt),
- Die vom LLM vorgenommene Klassifikation (als „Klassifikation“ aufgeführt) und
- Die Begründung des LLMs für seine Klassifikation (als „Erklärung“ aufgeführt).

### Inputdaten:
- **Klauseln**: Alle Klauseln, die in der letzten Iteration vom LLM falsch klassifiziert wurden, befinden sich unter ### Klauseln.
- **Aktuelle Klassifizierungskomponente**: Der aktuelle Prompt, der vom Klassifizierer verwendet wurde, ist ebenfalls bereitgestellt. Dieser Prompt
beinhaltet die „Klassifizierungskomponente“, welche beschreibt, unter welchen Bedingungen eine Klausel als „unzulässig“ und wann als „potentiell
unzulässig“ einzustufen ist.
- **Verteilung**: Tatsächliche Verteilung der Klauseln aus der Kategorie ###Kategorie### in die Bewertungskategorien ###available_labels###. Die
Verteilung ist als wahr anzunehmen, weil sie von professionellen Anwälten kommt.

### Deine Aufgabe:
Erstelle eine verbesserte Version der „Klassifizierungskomponente“, die den Klassifizierer so anleitet, dass die falsch klassifizierten Klauseln bei
der nächsten Iteration korrekt eingestuft werden. Die neue Komponente sollte:
1. Die spezifischen Bedingungen für die Klassifikationen einer Klausel als „unbedenklich“ und „potentiell unzulässig“ so präzisieren, dass alle
unter ### Klauseln gelisteten Fehler vermieden werden. Ein besonderes Augenmerk soll nicht auf den eindeutigen, sondern auf den Grenzfällen liegen.
Berücksichtige dabei auch die Erklärung des LLM-Klassifizierers für seine Bewertung.
2. Inkorporiere auch die Verteilung der Klauseln in die einzelnen Bewertungskategorien bei der Erstellung, allerdings nur indikativ, ohne die
konkrete Verteilung in Zahlen zu nennen.

Verfasse die neue Klassifizierungskomponente ohne unnötige Einleitungs- und Schlussformulierungen. Die Komponente soll direkt beginnen mit den
Kriterien und keine zusätzlichen Titel oder Erläuterungssätze wie „Die Klassifizierungskomponente wird wie folgt definiert“ enthalten.
Nutze das folgende Format, um die neue Komponente abzugrenzen:
<START>...Das ist ein Platzhalter für die neue Komponente...<ENDE>

### Klauseln: {clauses}
### Aktuelle Klassifizierungskomponente: {current_component}
### Verteilung: {distribution}

Passe die Komponente so an, dass alle bereitgestellten, falsch klassifizierten Klauseln in der nächsten Iteration vom Klassifizierer richtig
bewertet werden.
```

Figure 3.6.: Meta-prompt δ_c for refining the classification component. The placeholders {clauses}, {current_component}, and {distribution} are replaced with the set of misclassified clauses, the current component, and the distribution of clause classifications within the cluster, respectively.

Algorithm 7 *Refine_Component*$(\cdot)$ - line 13 in algorithm 1

Require: Candidate $c = (s, T)$ with component s and policies T, batch of clauses $B = \{b_1, b_2, \ldots, b_k\}$

Ensure: A new component s_{new} that incorporates the shortcomings of the current component s

1: Collect wrongly classified clauses that are fair or unfair:
$e = \{(x_i, y_i) \mid x_i \in B, \text{LLM}(P_c(x_i)) \neq y_i \wedge y_i \in \{0, 1\}\}$

2: Use e to create a new component that ought to compensate the shortcomings of the previous component:
$s_{new} = LLM_\delta(s, e)$

3: **return** s_{new}

Policy Refinement In the policy refinement step, we update the set of policies for each candidate based on the classification errors identified during the current iteration. This process specifically targets clauses labeled as void (2) that were misclassified by the LLM. Given a candidate $c = (s, T)$ with component s and policies T, and a batch of clauses $B = \{b_1, b_2, \ldots, b_k\}$, the refinement proceeds as follows:

1. Error Collection: During the classification process, errors are collected for all misclassified clauses in B. From this set of errors, we filter the clauses labeled as void.

2. Refinement Using Meta-Prompt: The filtered errors are fed into the policy refinement meta-prompt δ_p which instructs the LLM to generate a set of operations that specify how to modify the current policy set T to address the identified misclassifications. The refinement process is detailed in Alg. 8. In addition to the misclassified clauses, the meta-prompt includes the candidate's current set of policies. The LLM returns a set of operations, which can involve: 1. *Modifying an Existing Policy* or 2. *Adding a New Policy*. These operations are then applied to derive the updated policy set T_{new}.

```
Du bist ein Prompt Engineer, spezialisiert darauf, Prompt-Formulierungen für einen LLM-Klassifizierer zu verbessern. Der
Klassifizierer ist darauf spezialisiert, Klauseln der Kategorie {category} aus Arbeitsverträgen hinsichtlich ihrer rechtlichen
Zulässigkeit und Fairness zu bewerten und sie in einer der folgenden Kategorien zu klassifizieren: „unbedenklich“, „potentiell
unzulässig“ oder „unzulässig“.

Der Prompt besteht aus mehreren Teilen, einschließlich einer Liste von Regeln. Jede Regel beschreibt, unter welchen Bedingungen eine
Klausel als „unzulässig“ einzustufen ist.

Aktuell werden jedoch einige Klauseln, die von Anwälten als unzulässig bewertet wurden, vom LLM-Klassifizierer falsch klassifiziert.
Deine Aufgabe ist es, die Liste an Regeln so anzupassen, dass der LLM-Klassifizierer in der nächsten Iteration alle unzulässigen
Klauseln auch als solche erkennt.

Für jede unzulässige Klausel, die nicht als solche klassifiziert wurde, sind folgende Informationen angegeben:
- Die vom LLM vorgenommene Klassifikation (als „Klassifikation“ aufgeführt) und
- Die Begründung des LLMs für seine Klassifikation (als „Erklärung“ aufgeführt).
- Eine Liste von IDs der Regeln, die das LLM angewendet hat, um die Klausel zu klassifizieren (als Policy List aufgeführt).

### Unzulässige Klauseln, die falsch klassifiziert wurden:
{clauses}

### Aktuelle Regelmenge:
{policies}

Deine Aufgabe ist es, die Menge der Regeln so anzupassen, dass das LLM beim nächsten Durchlauf in der Lage ist, alle Klauseln
mithilfe der Gesamtheit der Regeln korrekt als unzulässig zu klassifizieren.

Zum Beispiel:
1) <START>Add() -> Eine Klausel, die in den AGB ein generelles Pfändungs- oder Abtretungsverbot für die Forderungen des Arbeitnehmers
festlegt, ist nach § 308 Nr. 9 lit. a BGB unwirksam und damit unzulässig<ENDE>

2) <START>Modify(352Fgdjsdgd) -> Eine Klausel, die in den AGB ein generelles Pfändungs- oder Abtretungsverbot für die Forderungen des
Arbeitnehmers festlegt, ist nach § 308 Nr. 9 lit. a BGB unwirksam und damit unzulässig<ENDE>
```

Figure 3.7.: Meta-prompt δ_p for refining the policy set based on misclassified void clauses. The placeholders {clauses} and {policies} are replaced with the set of misclassified void clauses and the current policy set, respectively.

Algorithm 8 *Refine_Policies*$(\cdot)$ - line 14 in algorithm 1

Require: Candidate $c = (s, T)$ with component s and policies T, batch of clauses $B = \{b_1, b_2, \ldots, b_k\}$

Ensure: A new set of policies T'_{new} that incorporate the shortcomings of the current policy set T'

1: Collect wrongly classified clauses that are void:
$e = \{(x_i, y_i) \mid x_i \in B, \text{LLM}(P_c(x_i)) \neq y_i \wedge y_i = 2\}$

2: Use e to create a new policy set that ought to compensate the shortcomings of the previous policies:
$T_{new} = \{T'_1, T'_2, \ldots T'_n\} = LLM_\delta(T, e)$

3: **return** T_{new}

Part 2 - Global Exploration via Candidate Crossover

In this step, we perform global exploration to prevent the candidate population from stagnating in local optima as shown in Alg. 9. To achieve this, we employ the evolutionary operator crossover, a technique widely used in genetic algorithms. While

Phase 1 focuses on refining candidates through local exploitation, this phase broadens the search space by combining elements of different candidates, enabling the discovery of diverse and potentially superior configurations.

Algorithm 9 *Crossover*$(\cdot)$ - line 17 in Algorithm 1

Require: Set of candidates $C = \{c_1, c_2, \ldots, c_m\}$, batch of clauses B, score function S
Ensure: A new candidate c_{new}

1: Compute the Hamming distance between all pairs of candidates in C:

$$\text{Hamming\_Distances} = \{(c_i, c_j, d_{ij}) \mid 1 \leq i < j \leq m,\ d_{ij} = \text{Hamming}(c_i, c_j)\}$$

2: Select the pair of candidates (c_a, c_b) with the greatest Hamming distance:

$$(c_a, c_b) = \arg \max_{(c_i, c_j, d_{ij}) \in \text{Hamming\_Distances}} d_{ij}$$

3: Retrieve the components and policies of c_a and c_b:

$$s_a, T_a \leftarrow \text{Component}(c_a), \text{Policies}(c_a)$$

$$s_b, T_b \leftarrow \text{Component}(c_b), \text{Policies}(c_b)$$

4: Perform crossover to create a new component:

$$s_{\text{new}} = \text{Crossover}(s_a, s_b)$$

5: Determine the candidate with the greater score:

$$c_{\text{best}} = \arg \max_{c \in \{c_a, c_b\}} S(c, B)$$

$$T_{\text{new}} = \text{Policies}(c_{\text{best}})$$

6: Create a new candidate:

$$c_{\text{new}} = (s_{\text{new}}, T_{\text{new}})$$

7: **return** c_{new}

The crossover process in our approach focuses on the component of the candidates while retaining the policies. By combining components from candidates with diverse characteristics, the search can escape local optima and explore new regions of the solution space which promotes the generation of prompt candidates with varied performance profiles, enhancing the likelihood of discovering high-performing solutions.

To select candidate pairs for crossover, we measure the Hamming distance between their performance vectors [110]. Unlike cosine similarity, which evaluates semantic similarity, Hamming distance assesses the number of differing positions in performance vectors. For instance, if a candidate classifies three out of five clauses correctly, its vector might be $[1, 1, 1, 0, 0]$. By comparing performance vectors in this way, we ensure that candidates with different error patterns are selected, fostering greater diversity in the resulting candidates.

```
Du bist ein Prompt Engineer, spezialisiert darauf, Prompt-Formulierungen für einen LLM-Klassifizierer zu verbessern. Der
Klassifizierer ist darauf spezialisiert, Klauseln der Kategorie ###Kategorie### aus Arbeitsverträgen hinsichtlich ihrer rechtlichen
Zulässigkeit und Fairness zu bewerten und sie in einer der folgenden Kategorien zu klassifizieren: „unbedenklich“, „potentiell
unzulässig“, „unzulässig“.

### Vorgaben:
1. **Klassifizierungskomponente A** und **Klassifizierungskomponente B**: Diese beiden Komponenten bieten jeweils eine spezifische
Grundlage für die Klassifikation und definieren die Bedingungen, unter denen eine Klausel als „unbedenklich“, „potentiell
unzulässig“, „unzulässig“ gilt. Beide Komponenten können unterschiedliche Schwerpunkte und Formulierungen aufweisen, jedoch sollen
ihre besten Elemente in der neuen Komponente zusammengeführt werden.

2. **Kombinationsstrategie**:
- Wähle die klarsten und präzisesten Anweisungen aus jeder Komponente und vereine diese so, dass eine umfassende, leicht
verständliche Anleitung entsteht.
- Berücksichtige dabei, dass die neue Komponente die Kernprinzipien beider Originalkomponenten beibehält, aber in einer optimierten,
strukturierten und widerspruchsfreien Form formuliert ist.
- Vermeide Redundanzen und streiche sich überschneidende oder widersprüchliche Inhalte.

### Klassifizierungskomponente A
###Klassifizierungskomponente A###

### Klassifizierungskomponente B
###Klassifizierungskomponente B###

### Aufgabe:
Erstelle die neue Klassifizierungskomponente aus Klassifizierungskomponente A und B und grenze sie mit den folgenden Tags ab.
Verfasse die neue Klassifizierungskomponente ohne Einleitungsformulierungen. Die Komponente soll direkt beginnen und keine
zusätzlichen Titel oder Erläuterungssätze wie „Die Klassifizierungskomponente wird wie folgt definiert“ enthalten.

<START>...Das ist ein Platzhalter für die neue Komponente...<ENDE>

Die neue Komponente soll die Kriterien aus beiden Klassifizierungskomponenten optimal vereinen, um die Genauigkeit der Klassifikation
in zukünftigen Iterationen zu maximieren.
```

Figure 3.8.: Meta-prompt for crossover, guiding the LLM to generate a new classification component by combining the best elements of two existing components. The placeholders {Klassifizierungskomponente A} and {Klassifizierungskomponente B} are replaced with the current components to be merged.

The crossover procedure is as follows: First, evaluate the Hamming distance between all pairs of candidates in the population based on their performance vectors. Second, identify the pair of candidates with the greatest Hamming distance. This ensures that candidates with the most distinct error patterns are chosen. Third, exchange components between the selected candidates to generate a new component using an LLM by instructing it with the meta-prompt shown in Fig. 3.8. The policies remain unchanged. Fourth, determine which of the two parent candidates has the higher score on the current batch. The new candidate adopts the policies of the candidate with the superior performance.

3.2.4. Phase 4: Selection

In the final step of each iteration, we select the candidates that will form the population for the subsequent iteration as shown in Fig. 3.9. Since our approach utilizes beam search, this selection process is equivalent to determining which candidates remain on the beam. The selection process evaluates all current candidates based on their scores on the batch of clauses and retains only the top-performing candidates. The procedure follows these key steps: First, each candidate in the population is evaluated using the score function S on the current batch of clauses B, unless this evaluation has already been performed. This produces a set of scored candidates. Second, the candidates are sorted in descending order of their scores. This ranking identifies the candidates that performed best on the current batch. Lastly, the top population $n_{population}$ candidates are selected to proceed to the next iteration. These candidates represent the best-performing configurations, ensuring that the beam search continues to refine high-quality solutions.

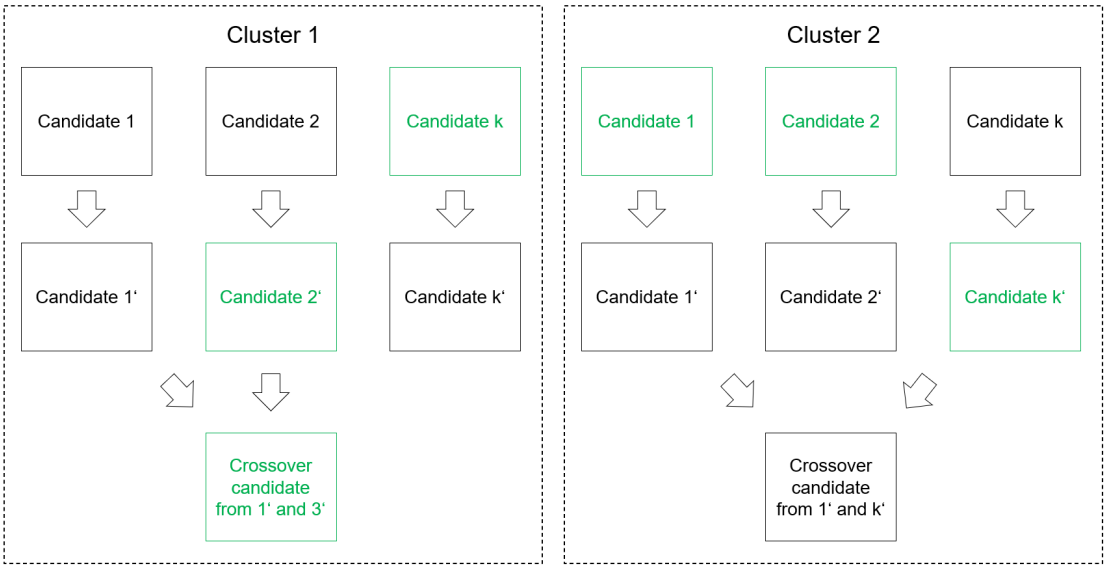


Figure 3.9.: Selection of top-performing candidates for the beam in two example clusters, where candidates are chosen based on their evaluation scores.

4. Experiment I – Testing the Efficacy of Automated Prompt Optimization

In this chapter, we introduce the setup (Section 4.1) and present the results (Section 4.2) of Experiment 1, which evaluates the performance gains achieved through APO in assessing the fairness of clauses in German employment contracts. To this end, we benchmark APO-optimized, category-specific prompts against a static, category-agnostic prompt crafted by domain experts, using a set of predefined metrics.

4.1. Setup

4.1.1. Experimental Conditions

This experiment evaluates two benchmark conditions using two models: *GPT-4o-mini* [125] (M_1) and *GPT-4o* [126] (M_2). The goal is to assess the performance improvement attainable by APO and how it is impacted by model capabilities. Each configuration is evaluated with both models (M_1, M_2), leading to four experimental conditions: (B, M_1), (B, M_2), (APO, M_1), and (APO, M_2).

Baseline (B):

This configuration serves as the baseline, representing classification quality in the absence of APO and providing a reference point to assess the impact of prompt optimization. In this setup, all 893 clauses are classified using a single static classification prompt (Figure 4.1), which embeds a structured reasoning process. The model is instructed to adopt the perspective of a lawyer, think step by step, and first assess whether any of the predefined policies apply. If a policy is applicable, the clause is classified as void; otherwise, the model evaluates its fairness. The static prompt includes all 20 initial policies (Table X).

Upper Bound (APO):

This configuration, which demonstrated the highest overall F1 improvement in preliminary experiments, represents the best-case scenario for performance enhancement when

```
<anweisung>
Du bist ein deutscher Anwalt, welcher auf Wirtschaftsrecht spezialisiert ist und Klauseln aus Arbeitsverträgen hinsichtlich ihrer
rechtlichen Zulässigkeit und Fairness bewertet.
Du denkst Schritt für Schritt und beachtest dabei die Regeln, welche dir zusammen mit der Klausel zur Verfügung gestellt werden.
Die Regeln stammen von professionellen Anwälten und sind somit immer als korrekt anzusehen.
Du bewertest eine Klausel nur dann als 'unzulässig', wenn sie gegen eine der gegebenen Regeln verstößt.
Wenn keine der gegebenen Regeln Anwendung findet, nimmst du eine eigene Einschätzung vor. Dabei bewertest du die Klausel aber nur
entweder als ‚potentiell unbedenklich' oder 'unbedenklich'. Du bewertest die Klausel als ‚potentiell unzulässig' wenn sie den
Arbeitgeber oder den Arbeitnehmer unangemessen benachteiligt oder besonders ungewöhnliche Formulierungen enthält.
Befolge bei deiner Antwort bitte das gegebene Antwortformat und gib an, welche Regeln du angewendet hast, um die Klausel zu
klassifizieren.
</anweisung>

<antwortformat>
    <bewertung>Gib hier die abschließende Klassifikation ein: 'unzulässig', 'potentiell unzulässig', 'unbedenklich'</bewertung>
    <regeln>Gib hier durch Kommas getrennt ausschließtlich die IDs derjenigen Regeln an, die du angewendet hast, um die Klausel zu
            klassifizieren. Wenn keine der Regeln Anwendung findet, dann bleibt die Liste leer</regeln>
    <erklaerung>Gib hier eine kurze Erklärung an, wie du zu deiner Bewertung gekommen bist</erklaerung>
</antwortformat>

<frage>
Ist die folgende Klausel unzulässig, potentiell unzulässig oder unbedenklich?
</frage>

<klausel>{clause}</klausel>

<regeln>
1: Pfändung/Abtretung

1.1: Pfändungs-/Abtretungsverbot nach § 308 Nr. 9 lit. a BGB in AGB ausgeschlossen. Kostenpauschale darf maximal 10 € betragen.

[Weitere Regeln]
</regeln>
```

Figure 4.1.: Static classification prompt employed for baseline conditions.

applying APO. It optimizes policies (PO = Yes) as well as the component (CO = Yes) of the prompt and clusters clauses on a category basis (Clustering = Category-based). Before optimization, clauses and policies are clustered according to predefined categories, such as Compensation or Vacation, as shown in Table 3.1. During optimization, a final classification prompt is tuned for each category by following the component and policy refinement procedure outlined in section 3.2.3 and 3.2.3 respectively: the component part of the prompt is refined by iteratively feeding errors from misclassified clauses that are fair or potentially void into the meta-prompt δ_c in Fig. 3.6, instructing the LLM to generate improved components. Policies are refined by leveraging the meta-prompt δ_p from Fig. 3.7 to address misclassifications of void clauses by modifying existing policies or adding new ones.

4.1.2. Evaluation Framework

To ensure an unbiased assessment of classification performance, we establish a standardized evaluation framework that enables direct comparability across the two experimental baseline conditions (B, M_1) and (B, M_2), as well as the upper-bound configurations incorporating APO (APO, M_1) and (APO, M_2)

Derivation of Final Performance Metrics for APO-Optimized Configurations

Unlike the baseline conditions (B, M_1) and (B, M_2), where performance metrics such as accuracy and F1-score can be directly computed from model outputs, the evaluation of APO-optimized conditions (APO, M_1) and (APO, M_2) necessitates an additional refinement step to ensure valid comparisons. The need for this step arises from the inherent structure of the APO process: selecting the top-performing candidate from each category based solely on its batch-level performance would introduce an upward bias in the estimated classification efficacy, as candidates are initially evaluated on a limited subset of clauses rather than the full category dataset.

To mitigate this bias and establish a fair evaluation paradigm, we re-assess all candidates from the final population—obtained after T iterations of APO—on the complete set of clauses within their respective categories. This ensures that batch-dependent fluctuations do not artificially inflate performance estimates. Within each category, the candidate c_i^F exhibiting the highest F1-score across the full category dataset is designated as the representative classifier for that category.

Following this selection, we compute a comprehensive suite of performance metrics by applying each representative candidate c_i^F to classify all clauses within its category.

Absolute Performance Metrics

For each experimental condition (C, M), where $C \in \{B, APO\}$ denotes either the *Baseline (B)* or *Upper Bound (APO)* configuration, and $M \in \{M_1, M_2\}$ represents either *GPT-4o-mini* (M_1) or *GPT-4o* (M_2), we compute a comprehensive set of performance metrics such as accuracy, recall, precision, etc.

$$\mathcal{M}(C, M) = \{\text{Accuracy}, \text{Precision}, \text{Recall}, \text{F1-score}\}$$

These metrics are evaluated at three levels:

- **Global Performance:** Aggregated metrics computed across all clause categories.
- **Category-Specific Performance:** Aggregated metrics computed across all clause categories.
- **Label-Specific Performance:** Per-class (fair, unfair, void) precision, recall, and F1-scores, measured both globally and at the category level.

Quantifying the Impact of APO

One of the key research objectives is to evaluate the effectiveness of APO in improving classification accuracy. To measure the impact of APO, we compute the relative

performance improvement from the Baseline to the APO-optimized Upper Bound configuration:

$$\Delta_{\text{APO}}(M) = \frac{\mathcal{M}(APO, M) - \mathcal{M}(B, M)}{\mathcal{M}(B, M)} \times 100\%$$

where$\mathcal{M}(B, M)$ represents the performance metric for the *Baseline* condition, $\mathcal{M}(APO, M)$ represents the performance metric for the *Upper Bound* condition and $\Delta_{\text{APO}}(M)$ quantifies the percentage change in performance attributed to APO for a given model M. This comparison is conducted separately for: *GPT-4o-mini* ($\Delta_{\text{APO}}(M_1)$) and *GPT-4o* ($\Delta_{\text{APO}}(M_2)$)

If APO demonstrates a substantial performance improvement ($\Delta_{\text{APO}}(M) > 0$), this suggests that prompt optimization plays a crucial role in improving classification accuracy. Moreover, differences between $\Delta_{\text{APO}}(M_1)$ and $\Delta_{\text{APO}}(M_2)$ can indicate whether higher-capacity models benefit more from APO than smaller models.

Quantifying the Impact of Model Choice

Another fundamental aspect of this study is assessing the role of model selection in classification performance. To measure the performance gap between *GPT-4o-mini* and *GPT-4o* we compute the relative difference between models under both baseline and APO conditions:

$$\Delta_{\text{Model}}(C) = \frac{\mathcal{M}(C, M_2) - \mathcal{M}(C, M_1)}{\mathcal{M}(C, M_1)} \times 100\%$$

where $\mathcal{M}(C, M_1)$ represents the performance of *GPT-4o-mini*, $\mathcal{M}(C, M_2)$ represents the performance of *GPT-4o* and $\Delta_{\text{Model}}(C)$ quantifies the relative performance gain** when upgrading from *GPT-4o-mini* to *GPT-4o*, under condition C. This metric is computed for both the baseline setup ($\Delta_{\text{Model}}(B)$) and the APO-optimized setup ($\Delta_{\text{Model}}(APO)$)

A positive $\Delta_{\text{Model}}(C)$ indicates that *GPT-4o* outperforms *GPT-4o-mini* under condition C. Additionally, if $\Delta_{\text{Model}}(APO) > \Delta_{\text{Model}}(B)$, it suggests that larger models benefit more from APO than smaller models.

4.1.3. Analysis of Optimization Efficacy and Stability in APO

To assess the dynamics of performance improvement during the Automatic Prompt Optimization (APO) process, we analyze the relative change in key classification metrics across successive refinement steps. This analysis provides insights into the effectiveness and stability of our propsed APO method over time, helping to determine whether

performance gains are sustained throughout the optimization process or subject to fluctuations.

Batch-Wise Optimization and Performance Characterization

As described in Chapter 3, our proposed APO method follows an incremental refinement approach in which candidates — each consisting of a *component* and a *set of policies* — are iteratively optimized within a population over multiple batch refinement steps. In each refinement step t, a set of candidate prompts competes within a batch of clauses. The best-performing candidate in a given batch, denoted as:

$$c_t^* = \arg\max_{c \in \mathcal{C}_t} \text{F1-score}(c)$$

is selected as the batch representative, where $\mathcal{C}_t$ represents the set of candidates in batch t and F1-score(c) is the classification performance of candidate c on its respective batch. Each batch representative c_t^* is characterized by its performance metrics (accuracy, F1-Score, ...) which reflect performance only on the batch used for optimization, rather than the full dataset.

Measuring Relative Improvement per Refinement Step

To track performance improvements over time we compute the relative change in each metric between successive refinement steps:

$$\Delta M_t = \frac{M_t - M_{t-1}}{M_{t-1}} \times 100\%$$

where M_t represents the metric value in refinement step t, M_{t-1} represents the metric value in the previous step $t-1$ and ΔM_t quantifies the percentage change in metric M from step $t-1$ to t, capturing whether performance continues to improve or stagnates.

Aggregating Performance Trends Across Refinement Steps

To obtain a global assessment of how performance evolves throughout the APO process, we compute the average relative improvement over all refinement steps T:

$$\overline{\Delta M} = \frac{1}{T} \sum_{t=1}^{T} \Delta M_t$$

This measure provides an estimate of the expected performance gain per iteration, offering insights into the overall efficiency of APO.

4.1.4. Hyperparameter

The following hyperparameters were employed to configure the Automatic Prompt Optimization (APO) process for experimental conditions (APO, M_1) and (APO, M_2):

Language Model

Both *GPT-4o (M1)* and *GPT-4o-mini (M2)* are used both as the *Classification Model*, responsible for classifying clauses, and as the *Optimization Model*, guiding the iterative refinement of components and policies via meta-prompts.

Optimization Iterations

The number of iterations T for the APO process was set to 2. This value balances the trade-off between allowing sufficient optimization steps and computational efficiency.

Candidate Population

Each cluster maintains a population of $n_{\text{population}} = 3$ candidates throughout the optimization process. This population size ensures sufficient diversity for exploration and refinement while remaining computationally manageable.

Batch Size

The batch size b was set to 25 clauses, with adjustments based on the number of clauses in each category cluster. If a category contained fewer than 25 clauses, the batch size was set to the total number of clauses in the cluster (e.g., b = 8 for a cluster with 8 clauses). For clusters with more than 25 clauses, batches were formed with a maximum size of 25. If the final batch contained fewer than 25 clauses, it was randomly supplemented with additional clauses from the same cluster to reach the full batch size (e.g., for a cluster with 35 clauses, the first batch contained 25 clauses, and the second batch consisted of the remaining 10 clauses plus 15 randomly selected clauses from the cluster).

4.2. Results

This section evaluates the performance of APO-optimized prompts against the static baseline across key metrics. We analyze absolute performance, APO's relative impact, and the effect of model choice, followed by an in-depth error analysis. Appendix A provides a detailed overview of the final optimized components and policies used in

GPT-4o classification prompts, while the full set for GPT-4o-mini is available in the accompanying GitHub repository. To illustrate the nature of APO-derived components and policies, Figure 4.2 presents the final prompt component and Figure 4.3 the generated policies respectively for the GPT-4o setting, both for the category *Termination*.

```
Eine Klausel gilt als 'unbedenklich', wenn sie klar und eindeutig formuliert ist, weder wesentliche Nachteile für
den Arbeitnehmer hervorruft noch über das übliche Maß hinausgeht, und mit rechtlichen und tariflichen Standards im
Einklang steht. Insbesondere:
• Kündigungsfristen und Kündigungsregelungen sollen die gesetzlichen Mindestvorgaben einhalten und ein gewisses
  Maß an Flexibilität bieten. Es darf keine einseitigen Vorteile ausschließlich für den Arbeitgeber geben. Solche
  Klauseln sind unbedenklich, wenn sie keine unangemessenen Belastungen für den Arbeitnehmer darstellen
• Flexible Regelungen wie kontrollierte automatische Beendigungsmechanismen und Altersgrenzen sind unbedenklich,
  sofern sie üblichen rechtlichen und tariflichen Standards entsprechen und nicht diskriminierend wirken.
• Bei der Abgeltung von Resturlaub ist es notwendig, dass die gesetzlichen Urlaubsansprüche vollständig
  respektiert werden. Klauseln in diesem Zusammenhang müssen transparent sein, damit sie unbedenklich sind.-
  Klauseln zur Probezeit sind zulässig, solange sie keine spezifischen Nachteile enthalten, wie übermäßige
  Kündigungsfristen.
• Die Gleichbehandlung beider Vertragsparteien in Kündigungsfristen ist unbedenklich, wenn keine unausgewogene
  Verlängerung der Verpflichtungen für den Arbeitnehmer besteht.

Eine Klausel wird als 'potentiell unzulässig' eingestuft, wenn sie:
• Formulierungen aufweist, die verschiedene Interpretationen zulassen könnten und dadurch eine potenzielle Last
  oder Benachteiligung für den Arbeitnehmer darstellen, ohne offen gegen rechtliche Standards zu verstoßen.-
• Praktiken beschreibt, die durch missverständliche oder mehrdeutige Formulierungen interpretiert werden könnten
  und dem Arbeitnehmer signifikante Nachteile ohne klare Schutzmechanismen bringen.
• Mechanismen beinhaltet, die der üblichen Kontrolle des Arbeitnehmers entzogen sind und ihm unverhältnismäßige
  Nachteile auferlegen könnten, wenn sie nicht fair vereinbart sind.

Die Klassifizierung soll stets im Einklang mit den gängigen arbeitsrechtlichen Standards und dem Schutz der
Arbeitnehmerrechte erfolgen, während der Einfluss der Klauseln auf das Gleichgewicht der Vertragsbeziehungen
gebührend beachtet wird.
```

Figure 4.2.: Component generated by APO for category *Termination*.

```
1. Eine Klausel, die während der Probezeit eine Kündigungsfrist zur ausschließlichen Bindung an das Ende eines
   Kalendermonats oder den 15. vorsieht, ist unzulässig, wenn nicht gleichzeitig eine taggenaue Kündigung möglich
   bleibt. Die Bindung an spezifische Stichtage ist jedoch als Zusatzregelung erlaubt, wenn (a) die gesetzlichen
   Mindestkündigungsfristen verbessert werden, oder (b) die Arbeitnehmer nicht unangemessen benachteiligt werden,
   wobei längere Kündigungsfristen grundsätzlich akzeptabel sind, sofern sie mit der gesetzlichen Regelung
   übereinstimmen.
2. Während der Probezeit darf die Kündigung nicht ausschließlich an feste Monats- oder Tagesfristen (z.B. nur zum
   15. oder Monatsende) gebunden sein, es sei denn, eine taggenaue Kündigung bleibt ausdrücklich möglich. Jede
   gegenteilige Regelung, die diese Flexibilität stark einschränkt, gilt als unzulässig.
3. Eine Klausel, die während der Probezeit nur sofortige Kündigungen erlaubt und keine taggenauen oder gesetzlich
   angemessenen Fristmöglichkeiten bietet, ist unzulässig. Es muss immer die Option einer flexiblen, taggenauen
   Kündigung entsprechend gesetzlicher Mindestanforderungen vorhanden sein.
```

Figure 4.3.: Policies generated by APO for category *Termination*.

4.2.1. Absolute Performance Metrics

Figure 4.4 presents the absolute performance metrics for the Baseline *(B)* and the APO-optimized *(UB)* configurations, evaluated using *GPT-4o-mini* (M_1) and *GPT-4o* (M_2). We want to highlight three key insights.

1. GPT-4o Outperforms GPT-4o-mini Across All Conditions

As expected, GPT-4o consistently achieves higher performance metrics than GPT-4o-mini, regardless of whether the classification is performed using the Baseline (B) or APO (APO) configuration. This is unsurprising given GPT-4o's higher model capacity, allowing it to generalize better and apply classification rules more effectively. When

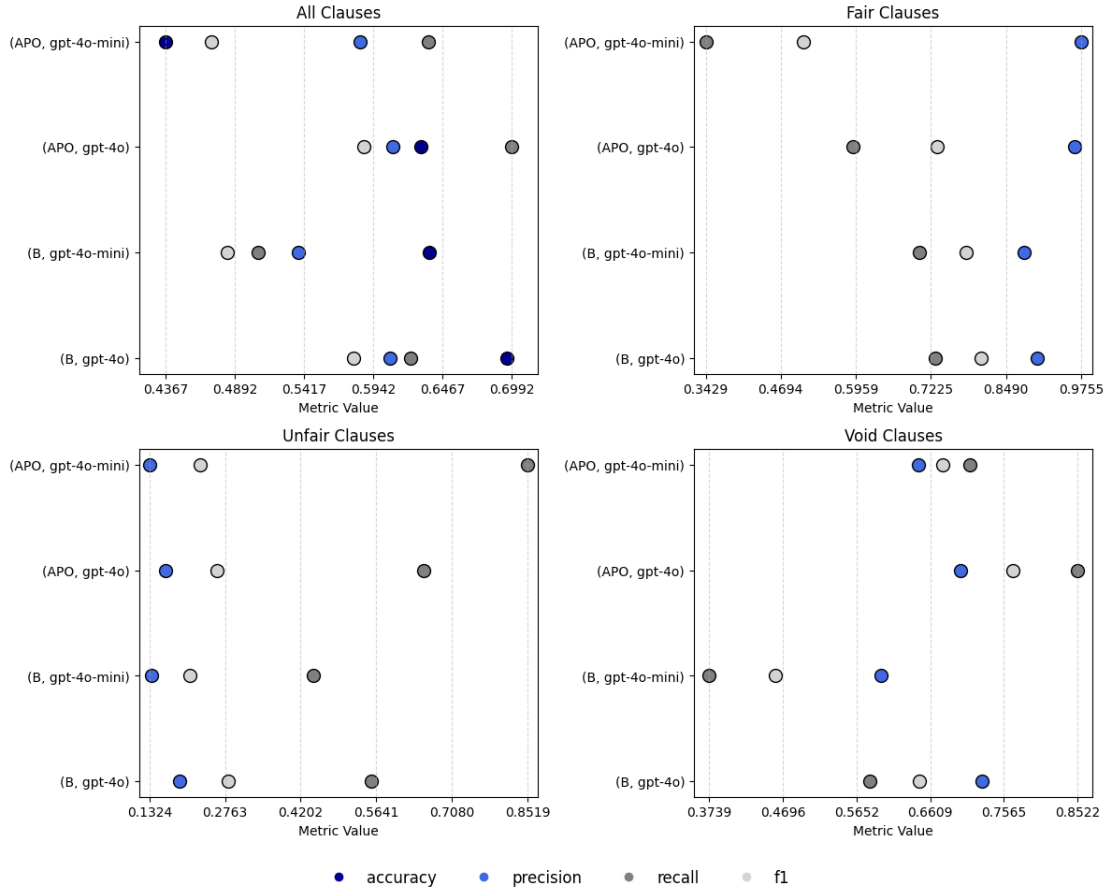


Figure 4.4.: Absolute classification performance metrics for all experimental conditions.

analyzing these metrics, we can make several high-level observations which we will elaborate on it following sections.

2. APO Improves F1-Score via Recall Gains but Reduces Accuracy

Compared to the Baseline, APO increases recall, precision, and overall F1-score, but at the cost of slightly lower accuracy. This trade-off suggests that APO enhances the

model's ability to correctly identify unfair and void clauses while slightly reducing the proportion of correctly classified fair clauses. This decrease in accuracy with APO can be largely attributed to its lower recall for fair clauses. Since fair clauses make up the majority of the clauses in the corpus (i.e., an imbalanced dataset), any decline in recall for this category has a disproportionate effect on overall accuracy. However, this decline does not indicate a failure of APO but rather a shift in classification behavior.

3. Higher F1 for APO is Driven by Recall Gains for Unfair and Void Clauses

The largest performance gains from APO occur in recall improvements for unfair and void clauses, which is a desirable outcome. The cost of false negatives for unfair and void clauses is significantly higher than false positives, as failing to identify problematic clauses can have legal consequences, whereas false positives only result in additional manual review.

4.2.2. Impact of Automatic Prompt Optimization

Figure 4.5 illustrates the relative impact of APO compared to the baseline for both GPT-4o and GPT-4o-mini. Based on these results, we can again make several observations.

1. APO Provides Only Marginal Gains Despite Iterative Optimization

The results indicate that APO yields only a slight improvement in overall classification performance, with an F1-score increase of just 2 percentage points (pp) for GPT-4o. However, this comes at the cost of a drop in accuracy by 9 percent, revealing a trade-off between classification sensitivity (recall) and correctness (accuracy). Given that APO systematically refines prompts over multiple iterations, a more substantial F1 improvement was expected, raising questions about the inherent optimization potential of APO in this task.

2. APO Benefits Void Clauses the Most While Compromising Fair Clause Recall

APO's impact is not uniform across clause types. The largest gains in recall are observed for void clauses, where APO increases identification rates substantially. This suggests that clear legal rule violations, such as those defining void clauses, are more amenable to rule-based optimization. However, these recall improvements come at the expense of a decline in recall for fair clauses, which represent the majority of the dataset. As a result, APO reduces overall accuracy.

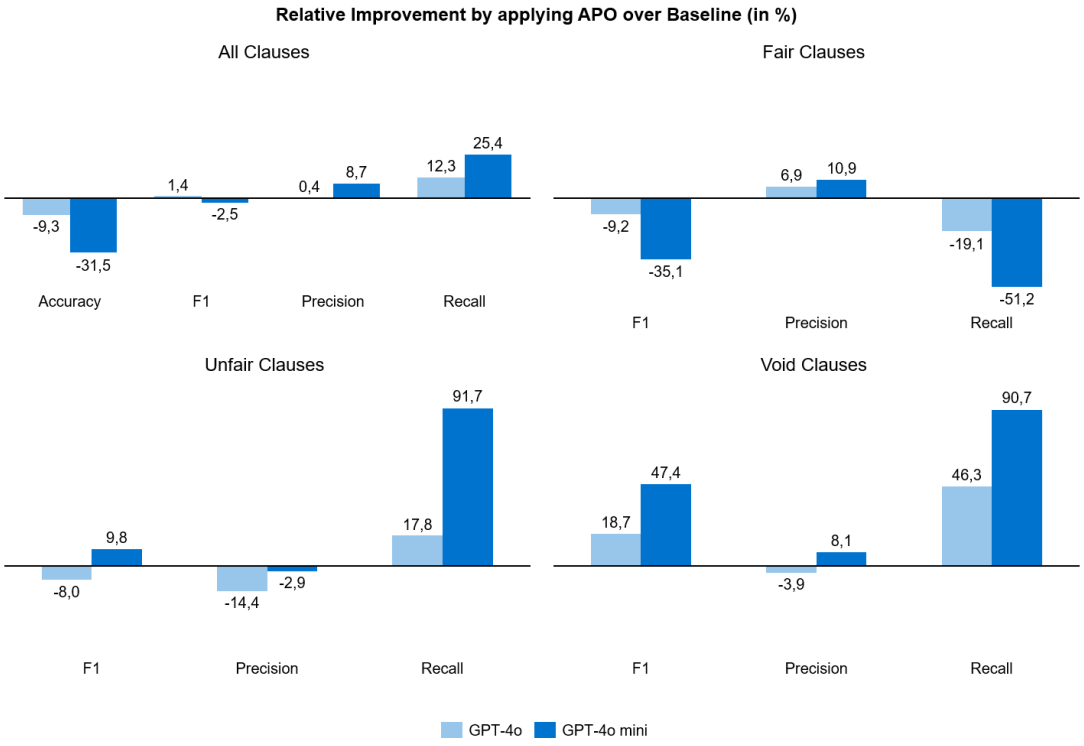


Figure 4.5.: Relative impact of performing APO compared to static baseline.

3. APO's Effect is More Pronounced for GPT-4o-mini, Indicating Higher Dependence on Prompt Optimization

Both GPT-4o and GPT-4o-mini exhibit similar directional trends in how APO influences their performance, reinforcing the consistency of APO's effects. However, GPT-4o-mini experiences more drastic performance shifts, implying that smaller models rely more heavily on external prompt optimization to achieve improvements. In contrast, GPT-4o appears more robust, suggesting that larger models inherently classify clauses more effectively, making them less sensitive to APO refinements.

4.2.3. Impact of Model Choice

Figure 4.6 illustrates the relative performance improvements obtained by using GPT-4o over GPT-4o-mini across classification metrics. The results consistently confirm that GPT-4o outperforms GPT-4o-mini, but the magnitude of improvement varies depending on whether the Baseline or the APO-optimized classification prompts were used.

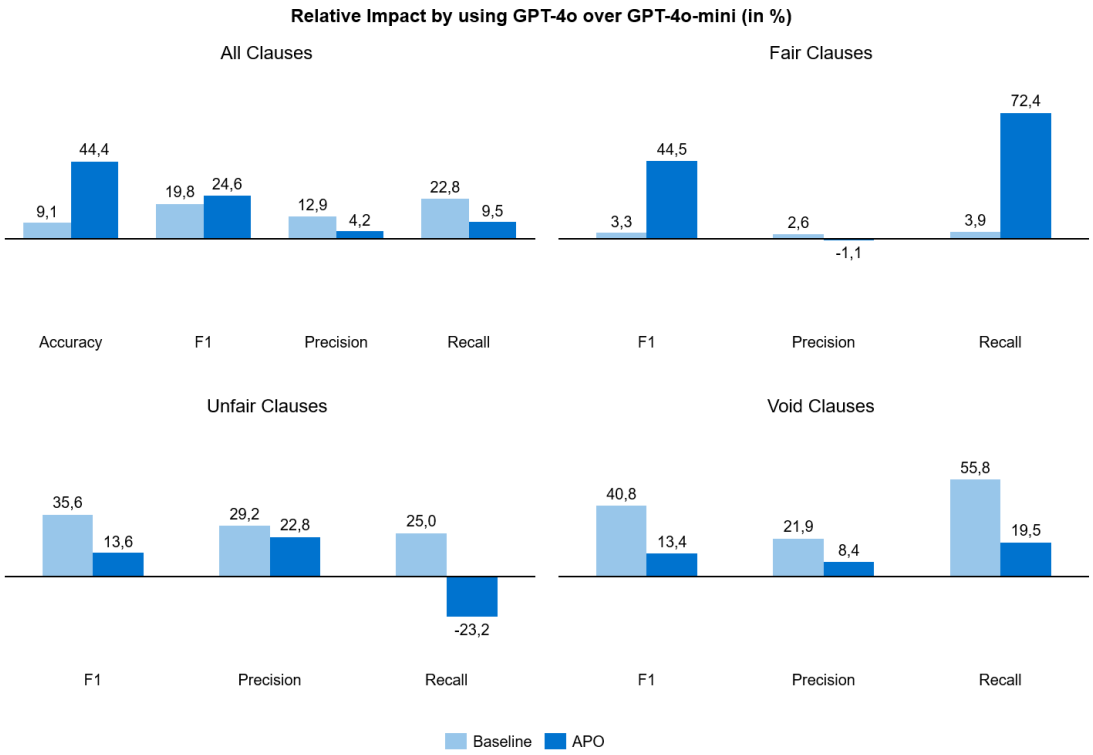


Figure 4.6.: Relative impact of GPT-4o over GPT-4o-mini using Baseline and APO.

1. GPT-4o Consistently Outperforms GPT-4o-mini Across All Configurations

Regardless of whether classification is performed using the Baseline prompt or the APO-optimized prompts, GPT-4o demonstrates higher performance across all key metrics (accuracy, precision, recall, and F1-score). This is unsurprising given GPT-4o's superior model capacity, which enables both better clause classification and more effective utilization of APO refinements.

2. GPT-4o's Gains Are Most Pronounced for Unfair and Void Clauses in the Baseline Condition

Compared to GPT-4o-mini, GPT-4o exhibits the largest relative improvements in recall and precision for unfair and void clauses when using the Baseline prompt. This suggests that GPT-4o's superior reasoning capabilities enable it to better interpret and apply the examination guides created by the lawyers. Additionally, the results indicate that GPT-4o has a stronger intrinsic understanding of what constitutes an unfair clause,

despite the vague definition of what makes up an unfair clause in the baseline prompt as confirmed by the higher recall/precision for unfair clauses.

4.2.4. Misclassification Distribution

Figure 4.7 compares misclassification patterns between Baseline and APO under the GPT-4o condition. The left matrix shows the error distribution for Baseline, with each cell representing the percentage of misclassified clauses per actual category. The middle matrix presents APO's error distribution, while the right matrix visualizes percentage point differences, with red indicating an increase and blue a decrease in misclassification rates.

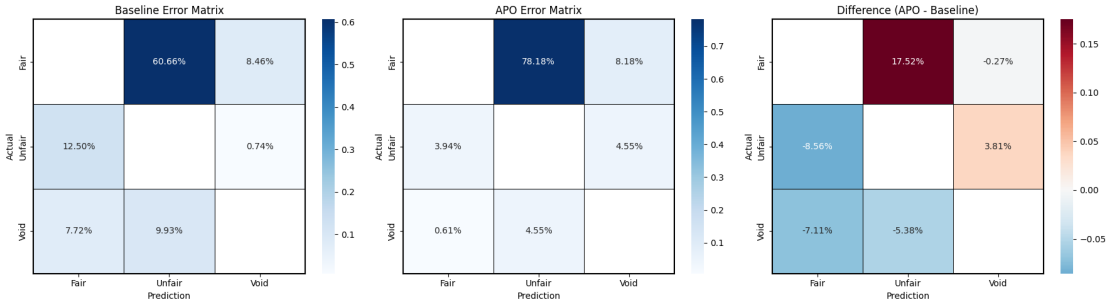


Figure 4.7.: Error matrices showing misclassification patterns of Baseline and APO.

The results reaffirm two key insights. First, APO adopts a more conservative classification strategy, reflected in a 17.52 percentage point increase in fair clauses misclassified as unfair. This suggests a systematic preference for minimizing false negatives in unfair and void clauses, even at the cost of higher false positives. Second, APO improves void clause recall, reducing their misclassification as fair. However, some void clauses are now labeled as unfair rather than void, indicating an improvement in flagging problematic clauses but not always in distinguishing between unfair and void.

Overall, this analysis quantifies APO's trade-offs. It enhances sensitivity to problematic clauses but increases over-flagging, particularly in borderline cases. APO shifts classification toward a risk-averse strategy, prioritizing conservative decisions without necessarily improving its understanding of fine-grained distinctions between fairness categories.

4.2.5. Optimization Efficacy and Stability in APO

Figure 4.8 presents the average improvement per update step in key classification metrics during the APO process using GPT-4o - configuration: *(APO, GPT-4o)*. It is

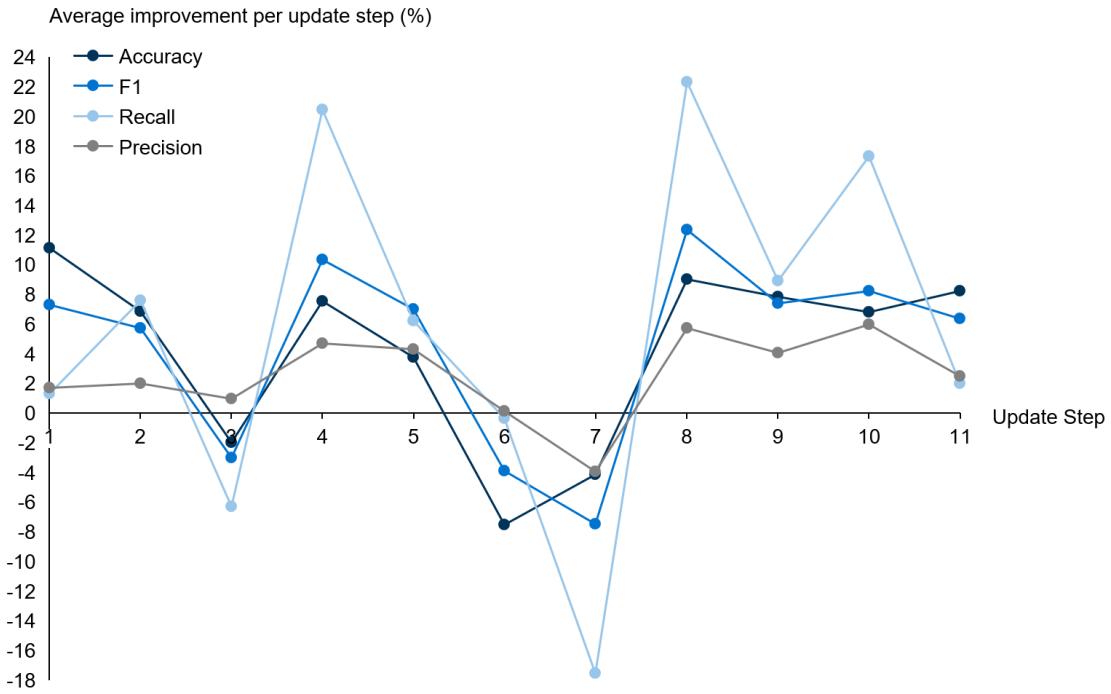


Figure 4.8.: Average improvement per update step during APO optimization.

important to note that while F1-score serves as the primary optimization metric, as described in Chapter 3, the evolution of accuracy, recall, and precision is not explicitly controlled during APO. Nevertheless, the overall trajectory suggests a relatively stable optimization process, where classification performance improves in most update steps. Across the optimization trajectory, only three update steps exhibit a deterioration in performance, while all others show consistent improvements. These results affirm the fundamental effectiveness of the APO approach, as confirmed by the average net improvement across all update steps (summarized in Table 4.1). However, the relatively

	F1-Score	**Accuracy**	**Recall**	**Precision**
Average Improvement	4.57	4.30	5.62	2.55

Table 4.1.: Improvement of performance metrics averaged across all update steps.

stable improvement trajectory during optimization raises a crucial question: Why do these improvements not translate into substantial performance gains in the final evaluation of optimized prompts? Figure 4.9, combined with a deeper understanding of

the APO process, provides a potential explanation. The figure reveals a strong negative correlation between the number of clauses in a category and the F1-score improvement achieved by APO relative to the Baseline—that is, the larger the clause category, the less effective the prompt optimization. This observation aligns with the batch-based nature of APO updates (Chapter 3). Since each update step refines the prompt using a fixed batch of 25 clauses, the iterative optimization process likely overfits to the clauses within a given batch, rather than generalizing improvements across the entire category. Consequently, while the APO process consistently improves classification performance within each batch-level refinement step, these improvements do not necessarily extend to the full category-level evaluation, resulting in diminishing returns as the category size increases.

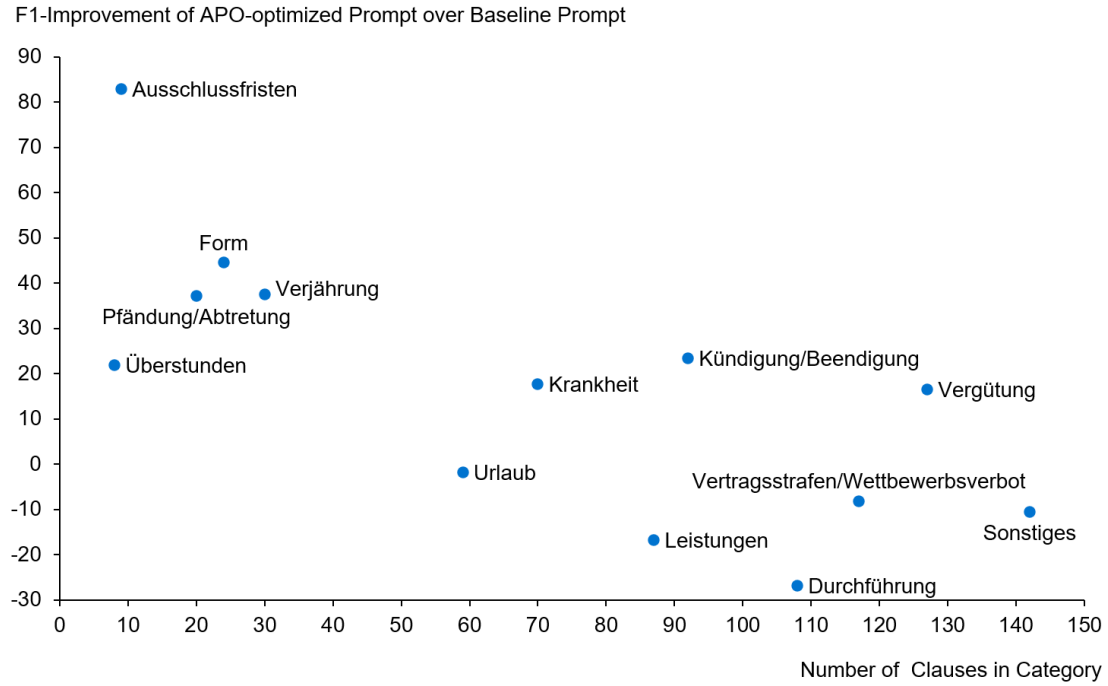


Figure 4.9.: F1-score improvement of APO-optimized prompts over the Baseline by clause category size.

5. Experiment II – Addressing APO Limitations with Redesigned Prompts

This chapter builds on the findings of Experiment 1 by addressing the limitations of APO and developing a revised prompt methodology. We begin by analyzing the shortcomings of APO observed in Experiment 1 (Section 5.1), followed by an introduction to our revised prompt methodology (Section 5.2). Finally, we present the setup (Section 5.3) and results (Section 5.4) of an experiment that benchmarks prompts generated using our revised methodology against both APO-optimized category-specific prompts and the static baseline prompt.

5.1. Why APO Falls Short: Motivation for a Redesigned Methodology

The Automatic Prompt Optimization (APO) algorithm, introduced in Chapter 3, was designed to iteratively refine prompts and enhance clause classification performance. However, results from Experiment I revealed only a marginal improvement in overall F1-score—approximately two percentage points—over a static prompt written by a legal domain expert. This stands in contrast to recent studies, where APO has demonstrated significant success in optimizing LLM-based classification tasks. To better understand this discrepancy, we identify several potential factors that may explain the limited optimization potential of the APO algorithm. These findings underscore the need for a redesigned methodology that addresses these limitations and enables more effective clause classification.

First, **ineffective feedback for prompt improvement** poses a significant challenge. When the model misclassifies a clause, it is unable to identify the underlying reasons for the error, leading to adjusments of the prompt that is often vague. This limitation is consistent with the findings of Ma et al. [122], who observed that LLM-based APO algorithms using reflection-based operators frequently generate generic feedback regardless of the specific classification errors. This outcome is intuitively understandable: if the model lacks the intrinsic knowledge or reasoning ability to classify a clause correctly in the first place, it is unlikely to identify the true cause of its error during

feedback generation. Instead, the model produces educated guesses based on its prior knowledge, which are disconnected from the actual misclassification. Consequently, this feedback fails to guide targeted improvements to the prompt components, stalling meaningful optimization and limiting the overall efficacy of the APO algorithm.

Second, the **ambiguous boundary between 'fair' and 'unfair' clauses** leads to frequent misclassifications. When the model encounters a clause that is clearly not void (as none of the policies apply), it must decide whether the clause is 'fair' or 'unfair.' However, the lack of a clear and specific explanation for what constitutes 'fair' or 'unfair'—a limitation rooted in the vague feedback issues discussed earlier—results in the model erring on the side of caution. In borderline cases, it often labels these clauses as 'unfair' to avoid potential false negatives. This conservative bias leads to 'fair' clauses being misclassified as 'unfair,' which is the most frequent failure mode for APO-optimized prompts. Given that 'fair' clauses make up the largest share of the dataset, this behavior explains the observed drop in accuracy compared to the baseline setting without APO.

Third, **stronger performance for 'void' clauses compared to 'unfair' clauses** highlights APO's reliance on explicit rule structures. 'Void' cases are inherently easier to classify due to the well-defined nature of their legal violations. However, another key factor contributing to their high F1-score post-APO is the presence of explicit policies that clearly outline the conditions under which a clause should be classified as void. These policies provide a structured decision framework that minimizes ambiguity. By contrast, 'unfair' clauses are more ambiguous and lack equally explicit decision criteria. The abstract nature of fairness makes it difficult to generalize classification rules, a challenge that APO was unable to overcome, as discussed earlier. This suggests that introducing explicit fairness criteria—similar to the policies used for void clauses—could improve the classification of unfair clauses.

5.2. Prompt Design Methodology

We suggest a new design for the prompts used to classify clauses as 'fair', 'unfair' or 'void' to address the limitations of APO identified in Section 5.1. Based on the acquired insights, we developed the following design principles for the revised classification prompts:

- **Category-Specific:** Preliminary experiments demonstrated that using a distinct classification prompt for each category yields superior results. This can be attributed to two primary reasons. First, category-specific prompts allow for precise tailoring of the conditions under which clauses are classified as 'fair,' 'unfair,' or 'void,' ensuring alignment with the unique characteristics of the

respective legal domain. Second, by excluding irrelevant rules and information, the prompts become less verbose, reducing cognitive load for the model [127, 128]. This streamlined structure enables the model to more effectively focus on the information critical for accurate clause classification.

- **No Iterative Optimization:** Experiment I demonstrated that the marginal benefits of APO in improving clause classification are highly limited, as discussed in Section 5.1. Consequently, in the revised design, we adopt static prompts that remain unchanged after their initial creation, eliminating the need for iterative adjustments.
- **Chain-of-Thought Reasoning Template:** To address the challenges of ambiguous boundaries between 'fair' and 'unfair' clauses and the inefficiency of feedback generation in misclassifications we design our new classification prompts as a Chain-of-Thought (CoT) reasoning template. Using this approach we instruct the model to derive classifications step by step by explicitly verifying all relevant rules, thereby enforcing a structured and logical decision-making process. Weit et al. [86] have shown that CoT templates significantly enhance LLMs' ability to solve complex, multi-step reasoning tasks by promoting systematic and interpretable outputs.
- **Default to Fair:** To avoid the challenge of explicitly defining what constitutes a 'fair' clause, we structure the classification prompt so that a clause is classified as 'fair' if and only if the model finds no justification in the preceding steps to classify it as 'unfair' or 'void.' Unlike the prompt design in Experiment I, the boundary between 'fair' and 'unfair' clauses is no longer explicitly explained in the prompt. Instead, similar to 'void' classifications, the specific conditions under which a clause should be classified as 'unfair' are explicitly listed. Only if none of these conditions apply should the model classify the clause as 'fair.'

To create classification prompts that meet the specified criteria for all categories, we applied a two-step approach (Figure 5.1). First, we manually developed a generic prompt template that adheres to the explained design criteria (Section 5.2.1). We then utilize this template in the second step, where we derive classification prompts for all categories in the clause corpus using a meta-prompt-based approach (Section 5.2.2).

5.2.1. Manual Design of a Prompt Template

To generate classification prompts for all categories, we first developed an abstract template manually, following the design principles outlined in Section 5.2. This

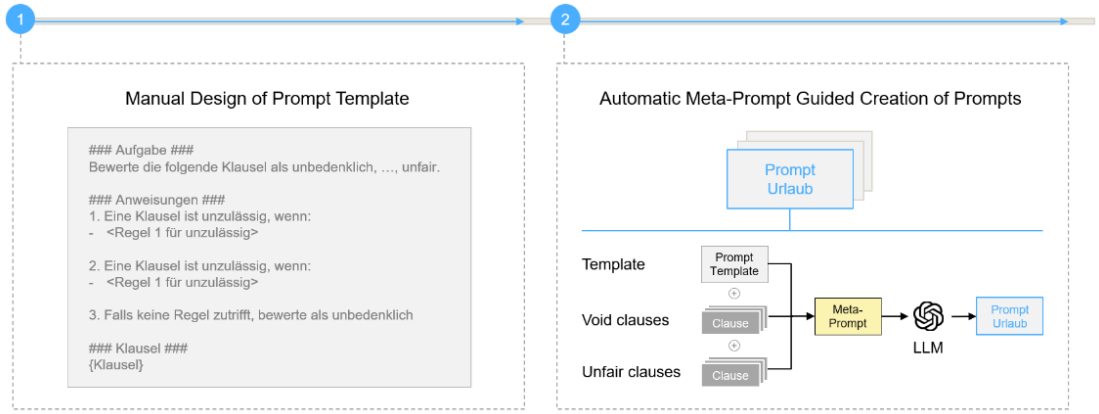


Figure 5.1.: Illustration of two-step process to derive category-specific prompts.

generalized prompt structure, shown in Figure 5.2, serves as the foundation for all category-specific classification prompts. It consists of the following core components:

- **Task Definition:** Clearly defines the task of classifying clauses as *fair*, *unfair*, or *void*.
- **Void Classification Rules:** Explicit and unambiguous conditions for identifying void clauses that constitute clear violations of legal provisions.
- **Unfair Classification Rules:** Criteria for clauses that are legally ambiguous or raise concerns but do not clearly constitute void provisions.
- **Default Classification Mechanism:** A clause is classified as *fair* ("unbedenklich") if and only if none of the specified rules for unfairness or voidness apply.
- **Response Format Instruction:** The model must select exactly one of the three predefined categories—*fair*, *unfair*, or *void*—without providing additional explanations.

5.2.2. Automatic Meta-Prompt Guided Creation of Classification Prompts

To generate the classification prompts for all categories (Table 3.1), we use a meta-prompt (Figure 5.3) that instructs the Creation LLM to produce a category-specific classification prompt. The structure of the meta-prompt is as follows:

- **Task Definition and Structural Consistency:** The meta-prompt instructs the *Creation LLM* to generate a classification prompt for a given category category,

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den angegebenen Regeln als
entweder "unbedenklich", "potentiell unzulässig" oder "unzulässig".

### Anweisungen ###
1. Eine Klausel ist 'unzulässig', wenn:
   - <Regel 1 für "unzulässig" basierend auf den bereitgestellten Beispielen>
   - <Regel 2 für "unzulässig" basierend auf den bereitgestellten Beispielen>
   ...

2. Eine Klausel ist 'potentiell unzulässig', wenn:
   - <Regel 1 für "potentiell unzulässig" basierend auf den bereitgestellten Beispielen>
   - <Regel 2 für "potentiell unzulässig" basierend auf den bereitgestellten Beispielen>
   ...

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als 'unbedenklich' bewertet
werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung vorliegt, entscheide dich
für 'unbedenklich‘.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere Erklärungen oder Text:
- 'unbedenklich'
- 'potentiell unzulässig'
- 'unzulässig'
```

Figure 5.2.: Generic template for deriving category-specific classification prompts.

ensuring that the structure and design principles established in the reference prompt remain unchanged.

- **Classification Prompt Template:** To maintain conceptual and procedural consistency across classification prompts, the meta-prompt includes the prompt template (Figure 5.2) as a baseline reference. The Creation LLM is explicitly instructed to preserve the structural logic and reasoning framework of this template when generating classification prompts for other categories.
- **Category-Specific Clauses for Rule Extraction:** The adaptability of the classification prompts is achieved through the injection of category-specific clauses. For each category, the meta-prompt supplies the unfair and void clauses of that category.
- **Rule Induction and Prompt Adaptation Instruction:** The meta-prompt explicitly instructs the Creation LLM to analyze the provided category-specific clause examples and induct the rules when a clause from the respective category should be classified as unfair or void.

By instructing the *Creation LLM* using this meta-prompt, we generate a distinct classification prompt for each category in the clause corpus.

```
### Aufgabe ###
Generiere eine neue Klassifikationsvorlage (Prompt) für die Kategorie "{category}". Diese
Vorlage soll denselben strukturellen Aufbau wie die unten vorgegebene allgemeine Template-
Struktur haben. Sie muss jedoch spezifisch auf die Anforderungen der neuen Kategorie abgestimmt
sein, indem sie Regeln aus den bereitgestellten Beispielen für *potentiell unzulässige* und
*unzulässige* Klauseln ableitet.

### Eingaben ###
1. **Potentiell unzulässige (unfair) Klauseln für die neue Kategorie "{category}":**
   {Unfair clauses}

2. **Unzulässige (void) Klauseln für die neue Kategorie "{category}":**
   {Void clauses}

### Allgemeine Template-Struktur ###
{classfication template}

### Antwort ###
```

Figure 5.3.: Meta-Prompt used for instructing the *Creation LLM* to induct category-specific classification prompt.

5.2.3. Applying the Framework: Example Prompt for Termination Category

To illustrate the application of our prompt design methodology, we present the final classification prompt generated for the category *Termination* in Figure 5.4. The classification prompts for all other categories were derived using the same meta-prompt-guided approach and are provided in Appendix B.

5.3. Experimental Setup

5.3.1. Experimental Conditions

This experiment introduces a new configuration, *Static (S)*, to assess the effectiveness of the redesigned classification prompts against the *Baseline (B)* and the *APO-optimized Upper Bound (APO)* from Experiment 1. Unlike Experiment 1, where two models (*GPT-4o-mini* and *GPT-4o*) were tested separately, Experiment 2 exclusively employs *GPT-4o* (used both as the *Creation LLM* and *Classification LLM*), given its superior performance in both prompt optimization and classification. As a result, model variability (M) is no longer a factor, and experimental conditions are solely defined by the classification configuration:

$$C \in \{B, UB, S\}$$

where *(B) Baseline* is the category-agnostic static prompt from Experiment 1, designed manually without optimization. *(APO) Upper Bound* is the set of category-specific

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den angegebenen Regeln als entweder
'unbedenklich', 'potentiell unzulässig' oder 'unzulässig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulässig', wenn sie eindeutig gegen arbeitsrechtliche Vorgaben verstößt. Das ist der Fall,
wenn:
• Die Kündigungsfristen für den Arbeitnehmer kürzer festgelegt sind als die gesetzlich vorgeschriebenen
  Mindestfristen oder wenn der Arbeitnehmer eine längere Frist als der Arbeitgeber einhalten muss.
• Während der Probezeit eine Kündigungsfrist vereinbart wird, die die gesetzlichen Vorgaben (i. d. R. 2 Wochen
  ohne Bindung an ein bestimmtes Monatsende) unterschreitet oder in unzulässiger Weise an ein Monatsende oder
  einen festen Stichtag geknüpft ist.
• Eine (teilweise) Erwerbsminderungsrente, Schwangerschaft, Behinderung o. Ä. zum automatischen Ende des
  Arbeitsverhältnisses führt, ohne dass die gesetzlichen Schutzvorschriften (z. B. Pflicht zur
  Weiterbeschäftigung) beachtet werden.
• Ein automatisches Ende des Arbeitsverhältnisses festgelegt wird, obwohl die Voraussetzungen des gesetzlichen
  Rentenalters bzw. eine wirksame Altersgrenzenregelung nicht eindeutig oder mit dem Gesetz vereinbar sind.
• Die Kündigung vor Arbeitsantritt ganz ausgeschlossen wird, obwohl das Gesetz eine Kündigung des noch nicht
  angetretenen Arbeitsverhältnisses nicht generell verbietet (möglicherweise aber erschwert).
• Das Recht auf fristlose Kündigung aus wichtigem Grund (gem. § 626 BGB) oder auf Kündigungsschutz (z. B. aus dem
  Kündigungsschutzgesetz) faktisch eingeschränkt oder ausgeschlossen wird.

2. Eine Klausel ist 'potentiell unzulässig', wenn sie problematisch oder unklar formuliert ist und möglicherweise
gegen die arbeitsrechtlichen Vorgaben zur Kündigung oder Beendigung verstößt. Das ist insbesondere der Fall, wenn:
• Eine Verlängerung der Kündigungsfristen zwar möglich ist, aber unklar bleibt, ob der Arbeitnehmer hierbei
  schlechtergestellt wird als der Arbeitgeber (z. B. längere Kündigungsfrist nur für den Arbeitnehmer).
• Regelungen zum automatischen Ende des Arbeitsverhältnisses bei Erreichen eines bestimmten Alters oder einer
  (teilweisen) Erwerbsminderung so ausgestaltet sind, dass nicht klar wird, ob eine gesetzeskonforme
  Weiterbeschäftigungspflicht oder ein Opt-out berücksichtigt wurde.
• In der Probezeit eine Kündigungsfrist vereinbart wird, die zwar länger als 2 Wochen ist, aber dennoch an starre
  Termine (z. B. Monatsende oder Quartalsende) geknüpft ist, ohne eindeutig gegen das Gesetz zu verstoßen, was
  jedoch zu Unsicherheiten führen kann.
• Eine Klausel zur Freistellung, zum Ruhen des Arbeitsverhältnisses oder zur Beendigung (z. B. beim Erreichen
  eines bestimmten Rentenalters) formuliert ist, aber Einzelheiten unklar bleiben und dadurch ein Konflikt mit
  den gesetzlichen Vorgaben möglich ist.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als 'unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung vorliegt, entscheide dich für 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere Erklärungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulässig'
   - 'unzulässig'

### Klausel ###
{clause}

### Antwort ###
```

Figure 5.4.: Final CoT classification prompt for the category *'Termination'*.

best-performing prompt generated through APO in Experiment 1 and *(S) Static* is the newly designed set of classification prompts that replace APO with a category-specific reasoning approach.

5.3.2. Evaluation Framework

To ensure comparability with Experiment 1, we compute the same performance metrics across all three configurations:

$$\mathcal{M}(C) = \{\text{Accuracy}, \text{Precision}, \text{Recall}, \text{F1-score}\}$$

where $C \in \{B, UB, S\}$. We do not re-evaluate (B) and (APO) in this experiment. Instead, we directly use the previously calculated metrics for $(B, \text{GPT-4o})$ and $(APO, \text{GPT-4o})$ from Experiment 1 as reference points.

5.3.3. Benchmarking the Performance of Static Prompts

To assess the effectiveness of the redesigned static classification prompts, we compute their relative performance improvement/degradation compared to the Baseline (B) and the APO-optimized Upper Bound (APO):

$$\Delta_{\text{S vs. B}} = \frac{\mathcal{M}(S) - \mathcal{M}(B)}{\mathcal{M}(B)} \times 100\%$$

$$\Delta_{\text{S vs. APO}} = \frac{\mathcal{M}(S) - \mathcal{M}(APO)}{\mathcal{M}(APO)} \times 100\%$$

where $\Delta_{\text{S vs. B}}$ measures how much (S) improves over or underperforms relative to (B) and $\Delta_{\text{S vs. APO}}$ quantifies the difference between the static approach (S) and the APO-optimized method (APO). A positive $\Delta_{\text{S vs. B}}$ would indicate that category-specific static prompts outperform the original baseline, demonstrating the benefits of structured reasoning and tailored decision criteria. Conversely, if $\Delta_{\text{S vs. APO}} < 0$, it would suggest that APO optimization still holds advantages over manually designed prompts.

5.4. Evaluation Results

The evaluation results of the benchmarking are summarized in Figures 5.5, 5.7, and 5.6. Figure 5.5 presents the absolute performance metrics for the static category-specific CoT prompts (S) compared to both the category-agnostic baseline prompt (B) and the APO-optimized classification prompts (APO). To facilitate a direct comparison, Figure 5.7 illustrates these results in relative terms, highlighting the magnitude of performance changes across configurations. Finally, Figure 5.6 provides an error matrix based on the classification results of the static CoT prompts (S), along with the absolute difference in misclassification patterns relative to the baseline prompt (B).

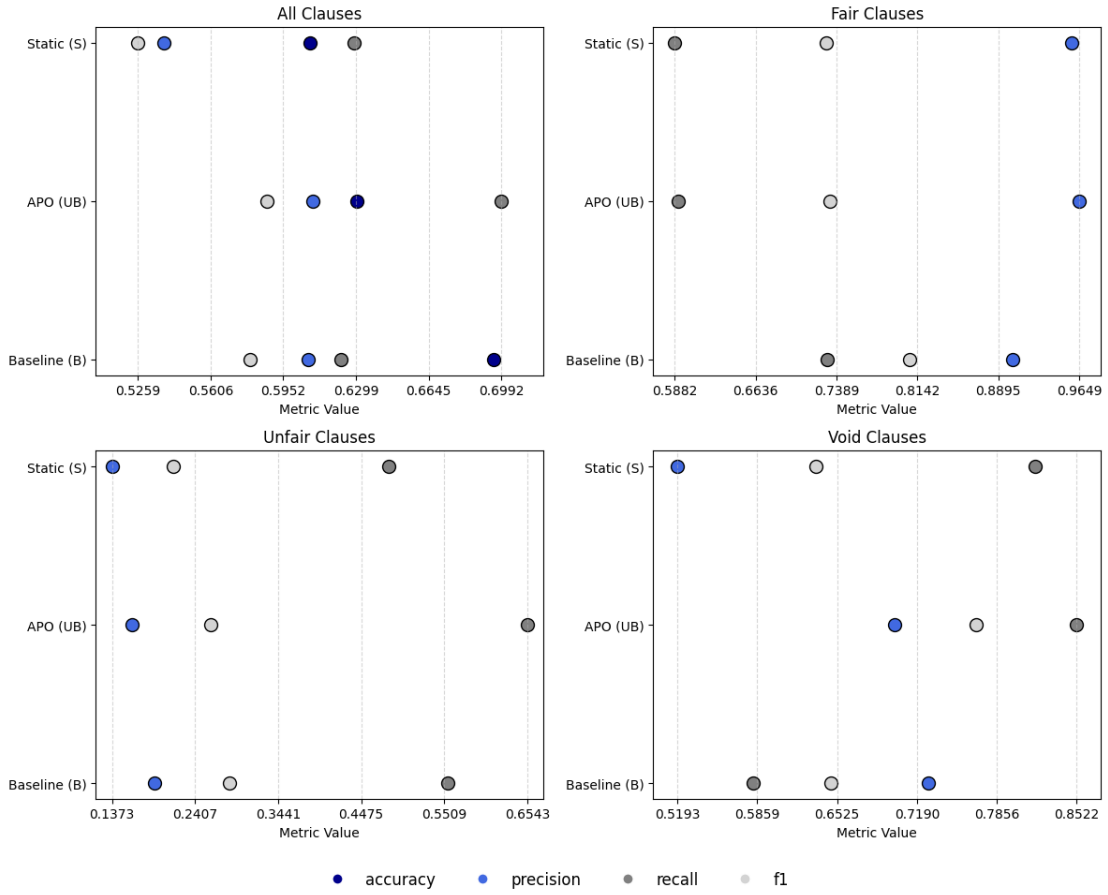


Figure 5.5.: Absolute classification performance metrics for category-specific static prompts compared against static category-agnostic baseline prompt and APO-optimized prompts.

1. Category-Specific CoT Prompts Underperform Compared to Baseline & APO

The absolute performance of the static category-specific prompts (S) is consistently lower than both the category-agnostic baseline prompt (B) and the APO-optimized prompts (APO) across all key classification metrics, including accuracy, F1-score, precision, and recall. This trend is also reflected in label-specific performance metrics, where deterioration is observed across all categories. The only exception is a notable 40% increase in recall for void clauses, as shown in Figure 5.7. These findings directly challenge the initial hypothesis that structuring clause classification as a chain-of-thought (CoT) reasoning problem with a default-to-fair mechanism would enhance

classification performance. Contrary to expectations, this approach not only fails to improve results over the APO-optimized prompts but also significantly underperforms compared to the category-agnostic baseline prompt.

2. Category-Specific CoT Prompts Achieve APO-Level Recall for Void Clauses but at the Cost of Precision

The static category-specific prompts (S) lead to a substantial improvement in recall for void clauses, detecting 40% more void clauses compared to the baseline prompt (B), which includes manually written examination guides by legal experts. Notably, the recall performance of (S) is only 5% lower than that of the APO-optimized prompts (APO), as shown in Figure 5.7. This finding has two key implications. First, the significant improvement over the baseline suggests that LLMs like GPT-4o can effectively infer the conditions under which a clause should be classified as void without requiring manually curated legal guides. This indicates that rule-based supervision by domain experts might not be necessary for capturing void clauses at a high recall level. Second, the relatively small gap between (S) and (APO) in recall highlights the diminishing returns of iterative policy refinement through APO. The majority of recall gains are already realized in the first policy derivation step, with limited additional benefit from further refinements. However, the significantly lower precision of (S) (25% below APO) underscores a key limitation: while static prompts improve recall, they also increase false positives, leading to more clauses being erroneously classified as void. This suggests that while LLMs can infer void classification rules, APO's iterative refinement remains critical for reducing misclassification errors and improving precision.

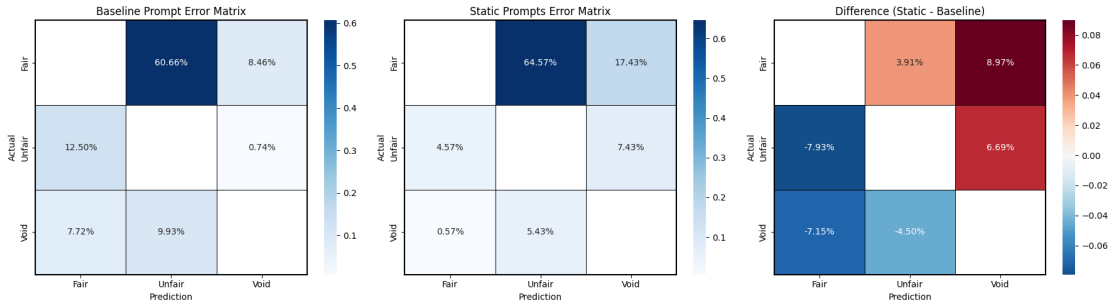


Figure 5.6.: Error matrices showing misclassification patterns of baseline and static Prompt as well as the resulting absolute percentage point difference.

3. Category-Specific CoT Prompts Fail to Improve Unfair Clause Classification, Reinforcing Conservative Bias

Classification performance for unfair clauses worsens significantly with static category-specific CoT prompts (S) compared to both the category-agnostic baseline (B) and APO-optimized prompts (APO). Recall and precision drop sharply, leading to a lower F1-score (Figure 5.7). Misclassification patterns (Figure 5.6) show that while fewer unfair clauses are misclassified as fair, errors increase overall, with more (i) fair clauses classified as unfair or void and (ii) unfair clauses classified as void. This shift indicates that the new prompt structure fails to correct APO's conservative classification bias. The findings contradict the hypothesis that structuring prompts as CoT reasoning—listing only explicit criteria for unfair and void clauses while defaulting others to fair—would remove the need to define the fair-unfair boundary. Instead, results suggest the model still struggles with this distinction and defaults to conservative classifications in ambiguous cases.

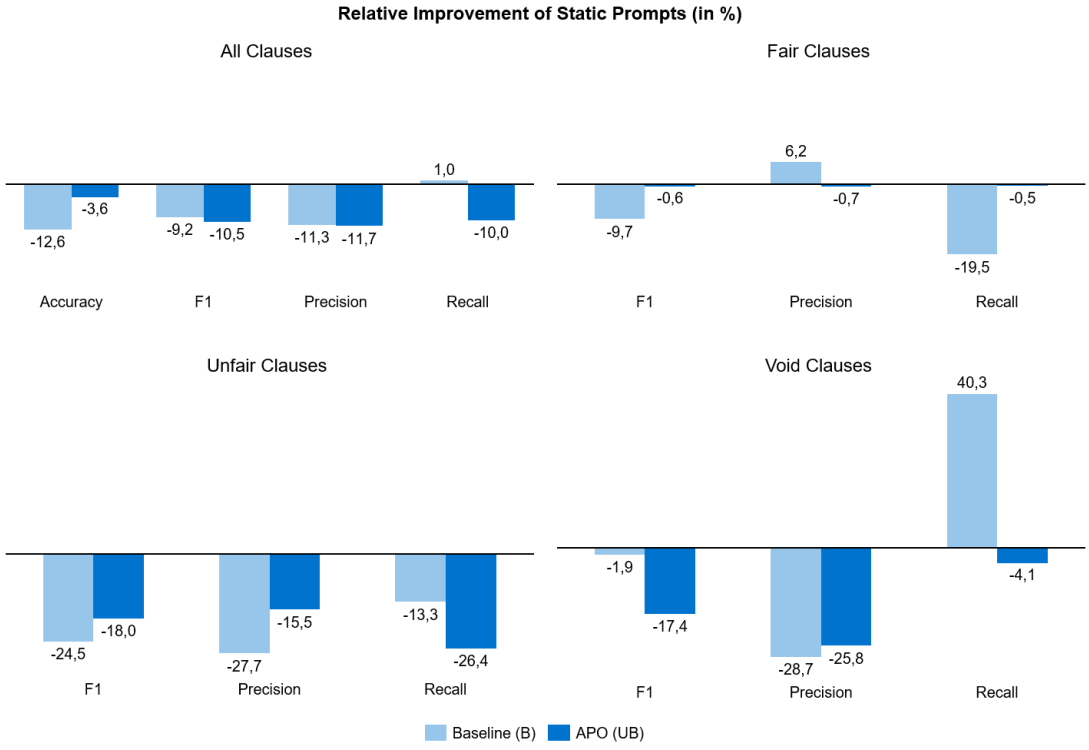


Figure 5.7.: Relative improvement (+) / deterioration (-) in performance of static category-specific prompts in comparison to APO and baseline

6. Discussion

In the following chapter, we contextualize the empirical findings we obtained from experiment I (Chapter 4) and experiment II (Chapter 5). We first outline how APO can be embedded into an information system framework to ensure its practical applicability (Section 6.1). Subsequently, we analyze the observed performance constraints, propose hypotheses to explain APO's limited upside (Section 6.2), and explore alternative strategies beyond APO for improving classification performance (Section 6.3).

6.1. Operational Integration of APO into an Information System Framework

The ultimate goal is to integrate the APO algorithm developed in this work into an information system such as KIBeKodA [11], enabling lawyers and legal professionals to systematically assess the fairness of employment contract clauses using an LLM-based classification framework. To facilitate the practical deployment of APO-driven clause evaluation, several key aspects of system integration must be considered. The following section outlines these critical factors.

6.1.1. System Workflow: From Clause Extraction to APO-Based Prompt Refinement

Sculley et al. [129] have raised awareness about the challenges of technical debt in machine learning systems, emphasizing that successful ML deployment requires more than just model training—it necessitates rigorous model management, continuous monitoring, and modularization to prevent performance degradation over time. This insight has contributed to the development of MLOps, which extends DevOps principles to machine learning by incorporating automated model retraining, deployment pipelines, data versioning, and performance monitoring to ensure model robustness and adaptability in real-world applications [130]. A key MLOps principle that is particularly relevant to the KIBeKodA system in the context of APO integration is continuous improvement and automated retraining. Applying this principle translates to dynamically refining the prompts used for clause classification in employment contracts through

our proposed Automatic Prompt Optimization (APO) algorithm. This approach offers two primary advantages:

1. **Alignment with evolving legal standards:** Legal frameworks and judicial interpretations are constantly evolving. By continuously updating the classification prompts, we ensure that the model's decision-making process remains aligned with the latest legal laws and precedents, a critical requirement for practical deployment in the daily workflow of legal professionals. This is particularly important given that language models inherently reflect a static snapshot of legal knowledge from their training data unless supplemented by mechanisms such as Retrieval-Augmented Generation (RAG) [131].

2. **Improved alignment with human expert judgment:** By iteratively refining classification prompts based on misclassified clauses and feedback from legal professionals, our system progressively reduces the gap between model-generated classifications and expert legal assessments. This follows a broader trend in machine learning research on aligning AI systems with human preferences and expert knowledge [31, 132].

In the following, we outline the complete workflow from the moment a lawyer uploads an employment contract to the system, through the classification process, and finally to the automatic refinement of classification prompts via APO updates. The workflow is also illustrated in Figure 6.1.

1. Clause Extraction and Categorization

The process initiates when a lawyer uploads a contract, triggering the automatic extraction of individual clauses. Each extracted clause is subsequently mapped to a predefined category (e.g., *Termination*), ensuring that classification aligns with the relevant legal policies and jurisdictional requirements. There are two principal approaches for assigning a clause to its respective category: first, leveraging similarity in the embedding space to identify the most semantically relevant category; second, utilizing a dedicated *Category Assignment Prompt* that allows the LLM to determine the appropriate category.

2. Clause Classification With Category-Specific Prompts

Once a clause has been assigned to a category, the corresponding classification prompt is retrieved from a centralized prompt library, which maintains the most up-to-date versions of prompts across all categories. The retrieved prompt is then applied to determine whether the clause should be classified as fair, unfair, or void. This classification

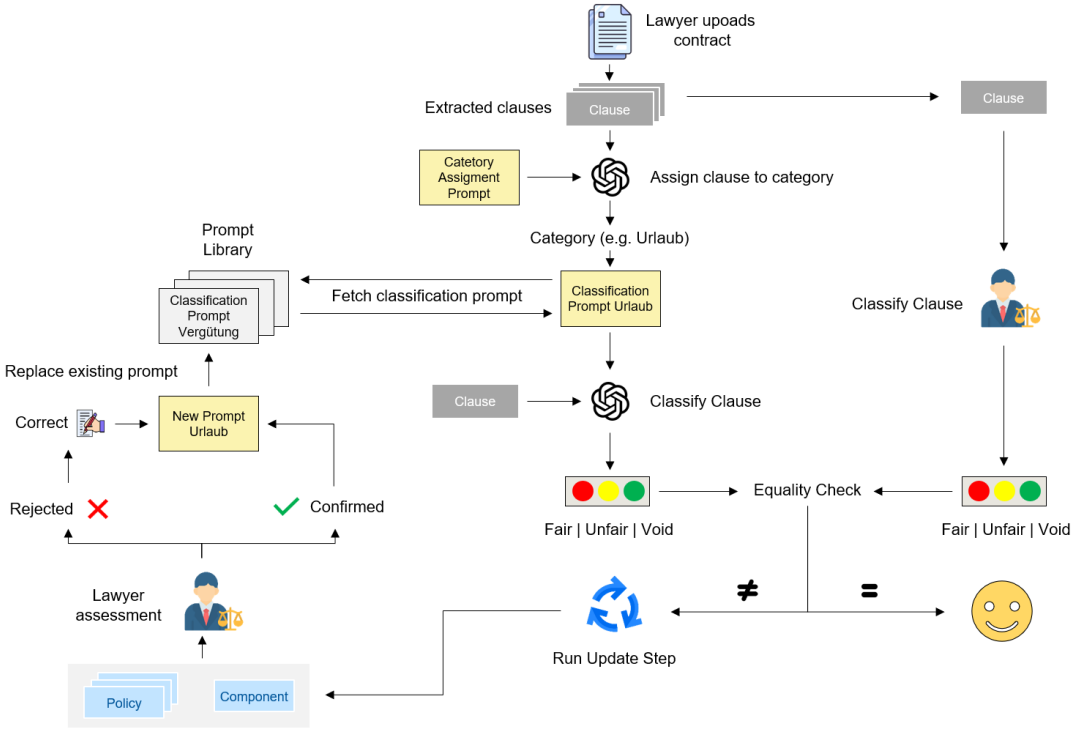


Figure 6.1.: MLOps worflow for self-improvement mechanism.

process is carried out by an LLM, leveraging the APO-optimized prompts introduced in Chapters 3 and 5.

3. Human-AI-Collaboration: Lawyer Validation & APO Updates

To ensure robustness and mitigate model biases, the system incorporates a human-in-the-loop validation mechanism [102, 133, 134]. The Equality Check compares the LLM's classification against a gold-standard assessment from legal experts. If both assessments match, the clause classification is accepted; otherwise, the discrepancy triggers an APO update step, where:

- A new optimized prompt is generated for the misclassified category by updating the respective component and policies.
- A legal expert reviews the suggested prompt updates.
- The expert accepts or rejects the modifications based on legal correctness and coherence. At this point, the legal expert can also refine the adjustment proposed

by the algorithm.

If confirmed, the new prompt replaces the existing version in the prompt library, ensuring prompts continuously evolve with the current state of the law.

6.1.2. Challenges and Future Considerations for Real-World Deployment

The challenge of aligning LLM outputs with legal expertise and lawyer expectations has been previously discussed. However, several critical obstacles must be addressed before deploying a system that assists lawyers in evaluating the fairness of employment clauses using APO-optimized classification prompts in a real-world production environment.

Transparency & Explainability:

Transparency and explainability are particularly critical in the legal domain, where decisions must be interpretable and justifiable [135, 136]. This is especially true for void clauses, which represent clear legal violations. To ensure accountability, it is essential to provide insight into which specific policy within the prompt led to a clause being classified as void. Achieving this requires a dynamically constructed prompt library, where policies relevant to each category are inserted into the classification prompt at runtime. These policies must follow a standardized data structure that enable *traceability* by identifying which policies were applied to specific clauses and *performance monitoring* by evaluating how well each policy performs in practice by tracking key metrics such as accuracy across the clauses it was applied to.

Legal Evolution Handling & Dynamic Reassessment

Legal norms continuously evolve due to newly enacted laws, judicial rulings, or regulatory changes. As a result, clauses previously classified as fair, unfair, or void may require re-evaluation to reflect the updated legal landscape. However, a static classification system lacks the ability to adapt dynamically, potentially leading to outdated assessments. To maintain legal accuracy, lawyers should periodically reassess previously classified clauses, particularly those in areas subject to frequent legal changes. If a reassessment leads to a different classification, this should automatically trigger an APO update step to refine the classification prompt accordingly.

Scalability & Prompt Expansion Management

As the number of classified clauses grows, classification prompts will inevitably expand, incorporating numerous policies, exceptions, and nuanced explanations. This can result

in excessively large and complex prompts. When prompts become too verbose, LLMs may struggle to extract and apply the relevant information efficiently, leading to classification errors, inconsistent reasoning, or longer inference times. To maintain scalability, we propose a periodic segmentation mechanism that clusters clauses using a clustering algorithm based on semantic similarity (e.g., K-Means). Instead of relying on a fixed number of expanding prompt, the system would increase the number of prompts, tuning a separate APO-optimized classification prompt for each cluster. This ensures that each prompt remains concise and contextually focused while maintaining consistency with evolving legal standards. Finding the optimal number of clauses each prompt is trained on is still an open question and an interesting avenue for future work.

6.2. Understanding APO's Limited Upside: Why Did the Algorithm Not Achieve Its Expected Improvement Potential?

While APO aimed to enhance clause classification performance in German employment contracts, the observed F1-score improvement of 2 pp fell short of initial expectations. However, it is important to highlight that the 45% increase in recall for void clauses represents a significant success. Given that undetected void clauses pose the highest legal and financial risks for both lawyers and affected employees, this improvement is a noteworthy outcome despite the marginal overall F1-score gain. Although we cannot provide a definitive explanation for the limited overall improvement, we propose hypotheses that may explain this outcome and suggest directions for future research to validate or refute these explanations. Two overarching macro-level hypotheses emerge:

1. **Data Limitations:** The size and composition of the clause dataset may have been insufficient for APO to effectively generalize decision boundaries, particularly in distinguishing between fair vs. unfair clauses and unfair vs. void clauses. Additionally, inconsistencies in human-labeled unfair clauses may have introduced noise, thereby reducing the theoretical upper bound of APO's optimization potential.

2. **Algorithmic & Model Limitations:** The design of the APO algorithm or the inherent capabilities of the underlying LLM may have constrained its ability to realize the expected performance gains. This could stem from limitations in the prompt refinement process, the LLM's inability to meaningfully self-reflect and improve classification logic, or fundamental constraints in how LLMs process complex legal reasoning.

In the following sections, we examine both data-related and algorithmic constraints in greater detail.

6.2.1. Algorithmic and Model Limitations

Despite the widespread success of Automatic Prompt Optimization (APO) in previous studies, our approach yielded only marginal improvements in F1-score. This raises a fundamental question: *Why do APO algorithms in works such as [38], [36] and [110] achieve substantial performance gains on standard benchmarks, while our implementation struggles to significantly enhance clause classification performance?* Two key explanations emerge as plausible.

Explanation 1: Expert-Designed vs. Randomized Prompt Initialization:

Many APO studies (e.g., [40] and [24]) initialize their optimization process with generic, task-agnostic prompts, such as *"Think step by step"* or minimal starting instructions. In contrast, our approach begins with a domain-expert-engineered prompt, incorporating legal examination guidelines and policies crafted by legal professionals. As a result, the potential improvement margin is inherently lower—optimizing an already well-structured, high-performance prompt leaves less room for APO to generate significant enhancements.

Explanation 2: Instruction-Level vs. Knowledge-Level Optimization:

A fundamental difference between our APO implementation and prior works lies in what aspect of the prompt is optimized. Standard APO benchmarks, such as those in the Big-Bench dataset [137], primarily focus on instruction-level optimization—modifying the way an LLM is instructed to perform a task (e.g., restructuring reasoning steps, changing few-shot examples, adjusting prompt phrasing).

By contrast, our APO framework fixes the instruction component of the prompt (e.g., *"You are a lawyer… classify the clause as fair, unfair, or void"*), instead optimizing the knowledge component—the legal rules and conditions that guide classification decisions. This distinction is crucial: instruction tuning primarily enhances task execution, whereas knowledge-level optimization requires the LLM to develop a deeper conceptual understanding of the legal domain.

However, research suggests that LLMs have fundamental limitations in self-reflection and knowledge refinement. Ma et al. [122] demonstrate that when APO algorithms generate self-improvement feedback, they frequently propose generic and repetitive modifications, failing to adapt adjustments to the actual misclassification error. This finding aligns with our empirical results: If the LLM lacks the legal knowledge to

classify a clause correctly in the first place, it is unlikely to possess the meta-cognitive ability to diagnose its own error and generate meaningful prompt refinements.

This suggests a systemic limitation of APO in knowledge-intensive domains such as law, medicine, and regulatory compliance. While APO excels at task-level performance tuning, its effectiveness in domains requiring deep factual and conceptual understanding remains questionable.

Experimental Validation of This Hypothesis

To empirically test whether LLM-based APO struggles with knowledge-level prompt optimization, a controlled experiment inspired by Ma et al.[122] could be conducted:

1. Generate an initial set of misclassified clauses using our APO-optimized prompts.
2. Trigger APO refinement, adjusting the prompts based on these errors.
3. Repeat the process, but this time randomly swap clause labels (fair, unfair, void), misleading the model with incorrect ground truth.
4. Compare the resulting APO adjustments: If the updated prompts show substantial overlap, it would confirm that the LLM fails to meaningfully differentiate between actual classification errors and arbitrary mislabeling, reinforcing the hypothesis that it lacks the ability to refine its own knowledge component effectively.

If validated, this would call into question the suitability of APO as a tool for optimizing LLMs in knowledge-intensive classification tasks, suggesting that alternative approaches, such as retrieval-augmented generation (RAG) or fine-tuning on expert-labeled legal cases, may be more effective in this domain.

6.2.2. Data Limitations

The limited improvement in F1 performance achieved through APO may be attributed to both quantitative and qualitative constraints of the underlying clause corpus. While quantitative factors primarily promote overfitting, qualitative deficiencies could fundamentally restrict the model's ability to learn meaningful decision boundaries in the first place.

Quantitative Deficiencies

A key limitation observed in Experiment I (Chapter 4) is the restricted sample size used during each APO update step. The optimization process refines prompts based on a limited subset of clauses, meaning that changes are made using only a fraction of the full

clause distribution. While this leads to apparent improvements in performance within the training batches, a different picture emerges when evaluating the final prompts on the complete dataset. When applied across all clauses in a category, APO-optimized prompts result in performance degradation, with only marginal improvements over the baseline prompt.

This outcome suggests that APO suffers from overfitting on batch-specific decision boundaries which is a common issue when performing supervised learning on small datasets [138]. Rather than learning generalizable classification patterns, APO updates tend to capture idiosyncratic features of the specific clause batches used in each update step. As a result, prompt refinements often fail to transfer effectively to unseen clauses, thereby limiting the overall generalization capacity of the optimized prompts.

A primary driver of this limitation is the insufficient dataset size and the imbalanced nature of the dataset [139]. The clause corpus contains numerous distinct legal provisions, each requiring nuanced classification criteria, which APO fails to capture consistently. This issue is particularly pronounced for unfair clauses, which make up only 9% of the dataset and vary significantly in their wording and contextual framing. Consequently, APO struggles to refine effective decision rules for distinguishing fair from unfair clauses. As a result, a conservative bias emerges as the default classification strategy. Since the distinction between fair and unfair clauses is inherently ambiguous and APO fails to optimize the classification threshold effectively, the model errs on the side of caution, frequently over-classifying borderline clauses as unfair or even void.

To mitigate these deficiencies, two approaches for expanding the dataset should be explored:

- **Natural Expansion:** Increasing the clause corpus by incorporating additional manually labeled clauses from real-world German employment contracts, ensuring a more balanced representation of all classification categories.
- **Synthetic Data Augmentation:** Generating additional unfair and void clauses through controlled perturbation techniques (e.g., paraphrasing, rule-based augmentation) to improve APO's ability to generalize across legal variations [140].

Qualitative Deficiencies

One potential explanation for the limited performance improvement of APO, particularly in distinguishing decision boundaries between fair, unfair, and void clauses, lies in inter-annotator agreement issues [73]. Discrepancies in legal experts' assessments of clause fairness raise fundamental questions about the objectivity of clause classification. This is especially pronounced for unfair clauses, which occupy a legal gray area between clearly fair and outright void clauses. Since unfair clauses neither constitute

an unequivocal legal violation nor an entirely permissible provision, their classification is inherently more subjective.

During the annotation process for the clause dataset on which APO was trained, disagreements between legal experts regarding the appropriate classification of certain clauses were frequently observed. This variability in annotations has direct implications for APO's effectiveness. If legal professionals themselves do not consistently agree on whether a clause is unfair, then APO—optimized on these labels—is effectively trying to learn from an unstable ground truth. Consequently, the algorithm may align with one annotator's interpretation but contradict another's, resulting in inconsistent decision boundaries. In this scenario, APO's capacity to improve classification performance is inherently constrained by the lack of agreement in its training data.

To assess whether inter-annotator agreement issues significantly impact the dataset, a blind labeling study could be conducted. In this setup, legal experts would independently reclassify a subset of the clause corpus without prior exposure to existing labels. Their classifications would then be evaluated using standard agreement metrics such as Cohen's Kappa or Krippendorff's Alpha [141]. If results indicate low agreement levels, this could suggest that the categorical scale (fair, unfair, void) is not sufficiently precise to capture the nuances of clause fairness.

A potential solution in such a case would be to transition from a categorical to an ordinal or continuous classification scale. Instead of rigidly forcing clauses into discrete categories, an ordinal scale (e.g., a 1–10 fairness score) could better represent the spectrum of unfairness, aligning more closely with real-world legal reasoning.

6.3. Beyond simple APO: Alternative Strategies for Clause Classification Improvement

Beyond APO, alternative approaches such as Ensemble Learning, Few-Shot Learning, Retrieval-Augmented Generation, and Parameter-Efficient Fine-Tuning offer additional avenues for improving the classification performance of fairness assessments in German employment contract clauses. The following sections explore these methods in greater detail.

6.3.1. Ensemble Learning

The first avenue worth exploring is the application of ensemble learning principles [142] to APO, specifically for the classification of clauses in German employment contracts. This could be implemented through either *Bagging* or *Boosting* as explored by previous studies [143, 144].

Bagging

One potential approach to mitigating APO's overfitting tendency while maintaining classification accuracy is Bagging (Bootstrap Aggregating) [145], a well-established ensemble learning technique that reduces variance by training multiple weak classifiers on different subsets of the data and aggregating their predictions. Instead of relying on a single optimized prompt per contract category, bagging would involve training a prompt ensemble, each tailored to different clause subsets. In this setup, multiple batches of clauses (e.g., 3-5) are sampled with replacement from the dataset for each contract category. For each batch, APO runs a limited number of update steps (1-3) to refine a batch-specific classification prompt, ensuring that different aspects of the clause distribution are captured. When classifying a new, unseen clause, the system applies each prompt in the ensemble independently, with the final classification determined through majority voting. By leveraging multiple independently optimized prompts, bagging introduces decision diversity, reducing the risk of overfitting to idiosyncratic patterns within specific clause batches.

Boosting

The principles of Boosting [146, 147] can also be adapted to APO. Unlike bagging, which aims to reduce variance by training independent classifiers, boosting iteratively improves weak classifiers by focusing on hard-to-classify instances. In the context of APO, boosting would involve dynamically adjusting the sampling probability of clauses based on their classification difficulty: After each APO update step, clauses that were misclassified (i.e., those that changed their label compared to human annotation) receive a higher weight in the next sampling iteration. This ensures that misclassified clauses are more likely to reappear in subsequent training batches. Since misclassified clauses repeatedly influence prompt updates, APO is pushed to adapt its decision boundaries specifically in areas where classification is challenging. Over multiple update steps, prompts should ideally become more robust by incorporating refined decision criteria derived from difficult clauses. While this approach may help APO better address decision boundaries between fair, unfair, and void clauses, it could also exacerbate overfitting. Since boosting prioritizes hard-to-classify instances, it may cause prompts to overly adapt to edge cases, at the expense of generalizability.

6.3.2. Few-Shot Learning

Few-shot learning presents another promising avenue for improving clause classification in German employment contracts beyond mere APO. While our approach in Chapter 3 has primarily focused on Instruction Optimization (IO), an alternative and potentially

complementary strategy involves Exemplar Selection (ES), where additional labeled clauses are included in the prompt to provide the model with concrete reference points. Few-shot prompting has been widely recognized as a powerful mechanism for guiding LLM behavior. Given the classification challenges previously discussed, integrating ES into the prompting strategy may offer significant advantages.

Selecting the right exemplars for few-shot classification is a non-trivial problem, as the effectiveness of in-context learning is highly sensitive to the choice and ordering of examples. Several established strategies from the literature provide potential pathways for optimizing exemplar selection in our domain. Similarity-Based Selection [96, 97, 98] could be particularly useful, as it ensures that the retrieved exemplars are semantically close to the clause being classified, reinforcing relevant patterns the model should recognize.

Another promising avenue is Influence Analysis, which quantifies the contribution of specific exemplars to model outputs [46, 102]. By systematically identifying clauses that strongly impact classification performance, the selection process can prioritize exemplars that enhance the model's decision-making on borderline cases. Given our findings on the conservative bias induced by APO, selecting exemplars that counteract this bias by including challenging but well-labeled edge cases may be helpful.

More dynamic approaches such as Model-Learned Selection [103, 104] or LLM-Generated Exemplars [101, 105, 106, 107] could also be explored. Instead of manually curating an exemplar bank, these approaches allow the model to either autonomously learn which clauses serve as the best exemplars or even generate its own high-quality reference cases. Such methods could mitigate issues related to dataset imbalances, particularly in underrepresented categories like unfair clauses.

6.3.3. Retrieval-Augmented Generation (RAG)

Beyond the limitations of APO, an alternative approach to improving the classification of employment contract clauses is Retrieval-Augmented Generation (RAG) [131]. RAG enhances language model performance by incorporating an external retrieval mechanism, allowing the model to supplement its parametric knowledge with dynamically retrieved information from a structured knowledge base. This method has been widely applied in domains requiring factual precision, domain adaptability, and interpretability.

In the legal domain, RAG has demonstrated its utility in a range of tasks. CBR-RAG [18] integrates case-based reasoning to enhance legal question-answering by providing contextually relevant precedents. LegalBench-RAG [19] establishes a benchmark for evaluating retrieval-based systems, emphasizing the accurate retrieval of legal text segments. Similarly, the Athena [20] framework employs RAG for legal judgment

prediction, achieving state-of-the-art results on the CAIL2018 dataset.

In the classification of employment contract clauses, RAG could serve as an alternative to APO by dynamically retrieving relevant legal information at inference time. Instead of relying on a static prompt containing predefined policies, the system would retrieve relevant sections from German labor law, judicial rulings, legal commentaries, or annotated contract clauses. These retrieved texts would then be appended to the classification prompt alongside the clause under evaluation, ensuring that the model's decision is grounded in the most relevant and up-to-date legal information. This approach could mitigate some of the shortcomings identified in APO, particularly in cases where the model struggles with ambiguous or evolving legal interpretations.

However, the effectiveness of RAG in this setting is contingent on the availability, structure, and quality of legal data sources. Inconsistent or outdated legal documents, retrieval errors, and conflicting legal interpretations could introduce new challenges.

6.3.4. Parameter-Efficient Fine-Tuning

Fine-tuning large language models is often impractical due to the high computational cost, the requirement for extensive labeled datasets, and the risk of overfitting to domain-specific nuances [2]. However, parameter-efficient fine-tuning (PEFT) techniques, such as Low-Rank Adaptation (LoRA) [28] and Adapters [148], provide a potential middle ground by modifying only a small subset of model parameters while preserving the pre-trained knowledge of the underlying model.

In the context of classifying clauses in German employment contracts, PEFT could be used to adapt an open-source LLM, such as Llama 3 [149], to better distinguish between fair, unfair, and void clauses. This process would involve curating a high-quality dataset of annotated employment clauses, freezing most of the model's layers while fine-tuning only task-specific components, and employing regularization strategies such as contrastive learning to improve the model's robustness against ambiguous borderline cases.

Despite its theoretical advantages, parameter-efficient fine-tuning is not encouraged as a primary alternative to APO in this setting. The effectiveness of fine-tuning depends heavily on high-quality training data, yet inconsistencies in legal expert annotations, as previously discussed, may limit its reliability. More importantly, unlike APO, which maintains explicit classification rules within prompts, a fine-tuned model would function as a black box, making it difficult to trace the reasoning behind its decisions—a critical drawback in legal AI applications. Additionally, employment law is inherently dynamic, requiring models to remain adaptable. Fine-tuned systems would necessitate periodic retraining to reflect new legal precedents, whereas prompt-based methods such as RAG or APO allow for immediate updates without modifying model weights.

7. Conclusion

This chapter concludes the thesis by summarizing our key findings and contributions, outlining limitations, and discussing how future research can build upon our results.

7.1. Summary & Contribution

In addressing Research Question 1 (RQ1), we conducted a comprehensive literature review on Automatic Prompt Optimization (APO), delineating its two primary paradigms: Instruction Optimization (IO)—which focuses on refining the instructional component of prompts to guide task execution—and Exemplar Selection, which involves selecting input-output examples to construct few-shot prompts. To further structure this landscape, we developed a set of criteria to classify IO approaches, including Optimization Space (Discrete vs. Continuous), Model Access (Black-Box vs. White-Box), and Optimization Technique (Gradient-Free vs. Gradient-Based). Beyond surveying conventional methods such as Soft Prompt Tuning, Reinforcement Learning, and Bayesian Optimization, we provided a formalized framework for LLM-based Optimizers, which have emerged as the dominant paradigm. Our analysis revealed that existing methods overlook a critical gap: optimizing prompts not just for instruction refinement but also for the implicit acquisition of domain-specific knowledge essential for task execution.

To bridge this gap and address Research Question 2 (RQ2), we developed a novel LLM-based APO algorithm specifically designed to acquire and refine legal domain knowledge for the fairness classification of clauses in German employment contracts. In our approach, we freeze essential components of the classification prompt while optimizing two key variable elements: (i) the Explanation Component, which defines the decision boundary between fair and unfair clauses, and (ii) the Policy Component, which encapsulates legal provisions under which a clause is deemed void. To further tailor prompts to specific legal subdomains, we adopted a category-level tuning approach, optimizing prompts separately for each clause category using an iterative optimization algorithm. The optimization process is driven by Beam Search for navigating the prompt search space, with iterative expansion enabled by local exploitation through reflection-based operators and global exploration via an evolutionary crossover operator selecting candidate prompts based on Hamming distance.

To evaluate our proposed APO algorithm and answer Research Question 3 (RQ3), we benchmarked the performance of category-specific, APO-optimized prompts against a static domain-expert baseline prompt on the dataset from [41] with three key findings:

1. The APO-optimized prompts achieved only a marginal improvement of 2 percentage points in global F1-score over the baseline, falling short of our expectations.

2. The APO-optimized prompts significantly improved the detection rate of void clauses by 45%, a critical success, as false negatives in void clauses represent the highest legal and financial risk for employees, employers, and reviewing law firms alike.

3. Despite demonstrating consistent performance gains at each step of the optimization process, these improvements did not consistently materialize in final evaluations across the full clause corpus of a given category. This suggests that the algorithm overfits to batch data, likely due to the imbalanced nature of the dataset, with only a small proportion of unfair and void clauses.

To contextualize these results, we formulated hypotheses to explain why the performance improvement potential of our algorithm and dataset appears inherently constrained, despite the substantial gains observed in other APO methodologies. Potential limiting factors include (i) data-related issues, such as insufficient clause volume and annotation inconsistencies (low inter-annotator agreement), as well as (ii) algorithmic and model limitations. We further outlined how the APO algorithm could be systematically improved within an information system through MLOps-driven continuous optimization and explored alternative methods beyond APO for enhancing LLM-based fairness classification, including ensemble methods, few-shot learning, retrieval-augmented generation, and parameter-efficient fine-tuning.

7.2. Limitations

Beyond the constraints outlined in Chapter 6, our findings must be considered in light of three key limitations.

Firstly, the research and thesis were conducted by computer science researchers, whose expertise lies in algorithmic development rather than legal scholarship. Given that the APO algorithm operates within the legal domain, this disciplinary gap introduced a knowledge deficit, particularly in evaluating the qualitative validity of the generated prompts and policies. While the researchers could assess the algorithm's performance through quantitative metrics and optimization criteria, they lacked the

legal expertise required to determine whether the derived prompts and policies aligned with existing legal frameworks and statutory requirements.

Secondly, the APO algorithm was trained on the entire clause corpus without a train-test split. This decision was necessitated by the limited dataset size—comprising only 900 clauses—and its inherent imbalance, with unfair and void clauses accounting for just 21% of the total corpus. Introducing a train-test split would have exacerbated overfitting tendencies, potentially preventing the algorithm from learning meaningful decision boundaries altogether. However, training without a separate test set limits the ability to assess the algorithm's generalization to unseen data, increasing the risk that observed performance improvements are due to memorization rather than true learning. This makes it difficult to determine whether the optimized prompts would remain effective when applied to new clauses beyond the training corpus.

Thirdly, we exclusively used OpenAI's GPT-4o [126] and GPT-4o-mini [125] for both clause classification and APO. A key limitation of this approach is that the effectiveness and efficiency of APO are highly sensitive to the choice of model, a phenomenon observed not only in our experiments but also substantiated by prior research. Consequently, we cannot determine whether performance improvements would have been more pronounced if alternative off-the-shelf models, such as Gemini 1.5 [150] or Llama 3 [149], had been used. Furthermore, it remains unclear whether fine-tuned models with legal domain knowledge, such as [16], would have been better equipped to internalize and apply the legal reasoning necessary for fairness classification in employment contracts.

7.3. Future Work

As potential avenues for future research have already been discussed in Chapter 6, this section provides a concise summary. The APO algorithm could be further refined by incorporating Bagging and Boosting mechanisms to enhance robustness and generalization. From a data perspective, conducting a second inter-annotator review and expanding the clause dataset with additional expert legal annotations—a step already planned [41]—could improve training quality. Additionally, building on the developed APO algorithm, an MLOps-inspired framework for continuous self-optimization could be implemented within a production system. Finally, it would be valuable to evaluate the APO-derived prompts in collaboration with legal domain experts to assess their substantive correctness and alignment with legal standards.

A. Additional Results of Experiment I

A.1. APO-optimized Classification Prompts

This section presents the final components and policies for all categories that were derived by the APO algorithm using GPT-4o as both the *Classification LLM* and the *Optimization LLM*.

A.1.1. Cut-off Periods

Component

Eine Klausel wird als 'unbedenklich' bewertet, wenn sie sowohl formal als auch inhaltlich den gängigen rechtlichen Standards entspricht, keine ungewöhnlichen oder irreführenden Formulierungen verwendet und die gesetzlich vorgeschriebenen Mindestanforderungen erfüllt. Insbesondere müssen die in der Klausel angegebenen Fristen für die Geltendmachung von Ansprüchen und die Erfüllung klar definiert sein und in Summe die vorgeschriebene Mindestdauer nicht unterschreiten.

Eine Klausel wird als 'potentiell unzulässig' klassifiziert, wenn sie zwar formal oder inhaltliche Mängel aufweist, wie zum Beispiel die Nutzung der Schriftform anstelle der gesetzlich geforderten Textform, die Regelungen jedoch nicht völlig gegen geltendes Recht verstoßen. Eine Klausel kann auch dann als 'potentiell unzulässig' angesehen werden, wenn sie zwar die gesetzliche Mindestfrist für Ausschlussfristen einhält, aber in Verbindung mit anderen Faktoren wie unklaren oder ungenauen Formulierungen rechtliche Unklarheit schafft. Falls im Rahmen einer zweistufigen Ausschlussfrist die gesetzlich geforderten Fristen eingehalten werden, jedoch die Voraussetzungen für die Erfüllung unpräzise oder nicht nachvollziehbar sind, kann die Klausel als 'potentiell unzulässig' betrachtet werden.

Policies

1. Eine Ausschlussfrist, die nur einseitig für den Arbeitgeber und nicht für den Arbeitnehmer gilt, ist unwirksam (BAG Urteil vom 21. Juni 2011 zum Az.: 9 AZR 203/10).

2. Eine Ausschlussfrist, die eine kürzere Frist als drei Monate ab Fälligkeit des Anspruchs vorsieht, ist unwirksam (BAG Urteil vom 28. September 2005 zum Az.: 5 AZR 52/05).

3. Eine Klausel, die eine Ausschlussfrist von weniger als drei Monaten festlegt, ist nach Auffassung des Bundesarbeitsgerichts (BAG) unangemessen (BAG Urteil vom 28. September 2005 zum Az.: 5 AZR 52/05).

4. Eine Klausel, die dem Arbeitnehmer in einer zweistufigen Ausschlussklausel für die Geltendmachung von Ansprüchen weniger als drei Monate in einer der Stufen einräumt oder die Vergleichsverhandlungen nicht berücksichtigt, ist unwirksam.

5. Eine Ausschlussfrist, die die Ansprüche des Arbeitnehmers auf Haftung bei Vorsatz, den gesetzlichen Mindestlohn oder unverzichtbare Rechte einschränkt, ist nach unwirksam (BAG Urteil Urteil vom 30. Januar 2019 zum Az.: 5 AZR 43/18).

6. Eine Klausel, die schriftliche statt textliche Form für die Geltendmachung von Ansprüchen verlangt, ist unwirksam, wenn sie nach dem Inkrafttreten der entsprechenden Rechtsprechungsänderung vereinbart wurde, da sie gegen § 309 Nr. 13 BGB verstößt. Sollte die Klausel jedoch als auffällig, aber potenziell zulässig betrachtet werden, wenn sie vor der Gesetzesänderung erstellt wurde und andere Fristanforderungen erfüllt sind.

7. Eine Klausel, die eine Ausschlussfrist festlegt, muss sicherstellen, dass mindestens zwei mal drei Monate (insgesamt sechs Monate) für die Geltendmachung und gerichtliche Durchsetzung von Ansprüchen zur Verfügung stehen. Klauseln, die genau diese Fristen einhalten, sind potenziell zulässig, sollten jedoch nicht sofort als unzulässig klassifiziert werden.

A.1.2. Execution

Component

Unbedenklich: Eine Klausel wird als 'unbedenklich' eingestuft, wenn sie klar und verständlich formuliert ist und den üblichen rechtlichen Standards und Praktiken des Arbeitsrechts ohne unangemessene Einschränkungen der Arbeitnehmerrechte entspricht. Solche Klauseln sollen die Transparenz und die Rechte der Arbeitnehmer gewährleisten, indem sie spezifische Details über Änderungen von Bedingungen wie Arbeitsorte, -zeiten oder organisatorische Strukturen enthalten, einschließlich geeigneter Mitbestimmungsrechte und Schutzmechanismen gegen Benachteiligungen. Vage Verweise auf externe Dokumente oder unklare Bestimmungen über Verpflichtungen sollten

vermieden werden. Ein Ausgleich zwischen Flexibilität für den Arbeitgeber und Schutz für den Arbeitnehmer muss sichergestellt sein, auch unter Berücksichtigung von Datenschutzregelungen wie der DSGVO.

Potentiell unzulässig: Eine Klausel wird als 'potentiell unzulässig' betrachtet, wenn sie Unklarheiten oder rechtliche Unsicherheiten aufwirft, insbesondere bei unbestimmten Vorgaben oder fehlenden klaren Spezifizierungen darüber, unter welchen Bedingungen Änderungen durchgeführt werden dürfen. Fehlt es an ausdrücklich formulierten Mitbestimmungsrechten oder Schutzmaßnahmen, entsteht ein Risiko der unangemessenen einseitigen Belastung des Arbeitnehmers. Auch unpräzise oder vage Verweise auf sich möglicherweise ändernde externe Regelwerke können eine Klausel als 'potenziell unzulässig' erscheinen lassen. Zudem sollte berücksichtigt werden, ob durch die Klausel übergeordnete Rechtsvorschriften missachtet oder interne Unternehmensregelungen übergangen werden könnten. Klare, präzise Formulierungen sind entscheidend, um Unsicherheiten zu minimieren und die Intention der Klausel eindeutig erkennen zu lassen.

A.1.3. Format

Component

Du bewertest die Klausel als 'potentiell unzulässig' wenn sie den Arbeitgeber oder den Arbeitnehmer unangemessen benachteiligt oder besonders ungewöhnliche Formulierungen enthält.

Policies

1. Eine Vorschrift, die eine strengere Form als die Textform für die Geltendmachung von Ansprüchen verlangt, ist unwirksam.

A.1.4. Illness

Component

Eine Klausel ist als 'unbedenklich' zu klassifizieren, wenn sie:

1. Keine übertriebenen Anforderungen stellt, die über die gesetzlichen Vorschriften hinausgehen, insbesondere in Bezug auf: - Die Vorlage von Arbeitsunfähigkeitsbescheinigungen, solange die gesetzlich vorgesehenen Fristen eingehalten werden und besondere Umstände berücksichtigt werden. - Die Art und den Umfang der Informationen, die dem Arbeitgeber über eine Krankheit mitgeteilt werden müssen,

beschränken sich auf allgemeine Anzeigepflichten ohne detaillierte Diagnoseoffenlegung. - Regelungen zu Vergütung und Urlaub, die gesetzeskonform und im Einklang mit tarifvertraglichen Vereinbarungen stehen.

2. Klar und konkret auf geltende gesetzliche oder tarifvertragliche Vorschriften verweist.

3. Keine übermäßigen Pflichten oder Rechte zugunsten des Arbeitgebers einseitig einführt und ausreichende Schutzmaßnahmen zur Wahrung des Gleichgewichts zwischen den Interessen des Arbeitgebers und des Arbeitnehmers vorsieht.

4. Alternative Lösungen oder Kompensationen für abgewandelte gesetzliche Ansprüche bietet, die für den Arbeitnehmer transparent und nachvollziehbar sind.

Eine Klausel ist als 'potentiell unzulässig' zu klassifizieren, wenn sie:

1. Anforderungen enthält, die im Kontext potenziell rechtswidrig erscheinen können, falls sie spezifische Umstände nicht adäquat berücksichtigen: - Zum Beispiel die Verpflichtung zur Untersuchung ohne konkrete Anhaltspunkte. - Pflicht zur Offenlegung von Krankheitsgründen, die die Diagnosen indirekt offenlegt.

2. Eine unangemessene Benachteiligung des Arbeitnehmers zur Folge haben könnte, indem sie: - Missverständnisse über gesetzliche Ansprüche verursacht oder nicht klar alternative Ansprüche bietet. - Gesetzliche Ansprüche ausklammert oder verändert, ohne ausdrückliche Kompensation. - Bestehende Versicherungsansprüche ohne eine transparente Lösung beeinträchtigt.

3. Verpflichtungen auferlegt, die zur übermäßigen Offenlegung sensibler Informationen führen, sofern nicht verhältnismäßige und berechtigte Interessen des Arbeitgebers existieren.

Bei der Bewertung einer Klausel sollte eine ausgewogene Abwägung zwischen den berechtigten Interessen beider Parteien erfolgen, um faire Arbeitsverhältnisse sicherzustellen.

Policies

1. Eine Verpflichtung, dem Arbeitgeber den genauen Grund für Abwesenheit (Diagnose/Erkrankung) mitzuteilen, ist nach unwirksam.

A.1.5. Termination

Component

Eine Klausel gilt als 'unbedenklich', wenn sie klar und eindeutig formuliert ist, weder wesentliche Nachteile für den Arbeitnehmer hervorruft noch über das übliche Maß hinausgeht, und mit rechtlichen und tariflichen Standards im Einklang steht. Insbesondere:

- Kündigungsfristen und Kündigungsregelungen sollen die gesetzlichen Mindestvorgaben einhalten und ein gewisses Maß an Flexibilität bieten. Es darf keine einseitigen Vorteile ausschließlich für den Arbeitgeber geben. Solche Klauseln sind unbedenklich, wenn sie keine unangemessenen Belastungen für den Arbeitnehmer darstellen. - Flexible Regelungen wie kontrollierte automatische Beendigungsmechanismen und Altersgrenzen sind unbedenklich, sofern sie üblichen rechtlichen und tariflichen Standards entsprechen und nicht diskriminierend wirken. - Bei der Abgeltung von Resturlaub ist es notwendig, dass die gesetzlichen Urlaubsansprüche vollständig respektiert werden. Klauseln in diesem Zusammenhang müssen transparent sein, damit sie unbedenklich sind. - Klauseln zur Probezeit sind zulässig, solange sie keine spezifischen Nachteile enthalten, wie übermäßige Kündigungsfristen. - Die Gleichbehandlung beider Vertragsparteien in Kündigungsfristen ist unbedenklich, wenn keine unausgewogene Verlängerung der Verpflichtungen für den Arbeitnehmer besteht.

Eine Klausel wird als 'potentiell unzulässig' eingestuft, wenn sie:

- Formulierungen aufweist, die verschiedene Interpretationen zulassen könnten und dadurch eine potenzielle Last oder Benachteiligung für den Arbeitnehmer darstellen, ohne offen gegen rechtliche Standards zu verstoßen. - Praktiken beschreibt, die durch missverständliche oder mehrdeutige Formulierungen interpretiert werden könnten und dem Arbeitnehmer signifikante Nachteile ohne klare Schutzmechanismen bringen. - Mechanismen beinhaltet, die der üblichen Kontrolle des Arbeitnehmers entzogen sind und ihm unverhältnismäßige Nachteile auferlegen könnten, wenn sie nicht fair vereinbart sind.

Die Klassifizierung soll stets im Einklang mit den gängigen arbeitsrechtlichen Standards und dem Schutz der Arbeitnehmerrechte erfolgen, während der Einfluss der Klauseln auf das Gleichgewicht der Vertragsbeziehungen gebührend beachtet wird.

Policies

1. Eine Kündigungsfrist, die nach der Probezeit ausschließlich taggenau nicht vorgesehen ist, sondern nur zu bestimmten Stichtagen, ist unbedenklich, wenn sie den gesetzlichen Regelungen entspricht oder diese verbessert und dabei den Arbeitnehmer nicht unangemessen benachteiligt.

2. Eine Klausel, die während der Probezeit eine Kündigungsfrist zur ausschließlichen Bindung an das Ende eines Kalendermonats oder den 15. vorsieht, ist unzulässig, wenn nicht gleichzeitig eine taggenaue Kündigung möglich bleibt. Die Bindung an spezifische Stichtage ist jedoch als Zusatzregelung erlaubt, wenn (a) die gesetzlichen Mindestkündigungsfristen verbessert werden, oder (b) die Arbeitnehmer nicht unangemessen benachteiligt werden, wobei längere Kündigungsfristen

grundsätzlich akzeptabel sind, sofern sie mit der gesetzlichen Regelung übereinstimmen.

3. Während der Probezeit darf die Kündigung nicht ausschließlich an feste Monats- oder Tagesfristen (z.B. nur zum 15. oder Monatsende) gebunden sein, es sei denn, eine taggenaue Kündigung bleibt ausdrücklich möglich. Jede gegenteilige Regelung, die diese Flexibilität stark einschränkt, gilt als unzulässig.

4. Eine Klausel, die während der Probezeit nur sofortige Kündigungen erlaubt und keine taggenauen oder gesetzlich angemessenen Fristmöglichkeiten bietet, ist unzulässig. Es muss immer die Option einer flexiblen, taggenauen Kündigung entsprechend gesetzlicher Mindestanforderungen vorhanden sein.

A.1.6. Tasks

Component

Kriterien für 'unbedenklich': - Die Klausel muss spezifisch und mit minimalem Interpretationsspielraum formuliert sein. Erfordert sie vom Arbeitnehmer Flexibilität, Mehrarbeit oder Anpassungen der Arbeitsbedingungen, so müssen Umfang, Dauer und Häufigkeit klar und detailliert spezifiziert sein. Gesetzliche Vorgaben und Branchenstandards sind dabei stets zu integrieren. - Jede Veränderung in Bezug auf Arbeitsort, Aufgaben oder Tätigkeiten muss transparent dargelegt werden und im Einklang mit dem bisherigen Tätigkeitsschwerpunkt des Arbeitnehmers stehen. Es ist sicherzustellen, dass die Zumutbarkeit solcher Änderungen offenkundig und verständlich begrenzt ist. - Anforderungen zur Berichterstattung oder sonstige Verpflichtungen müssen konkret und in spezifische, nachvollziehbare Szenarien eingebettet sein, ohne grundlegende Arbeitnehmerrechte zu beeinträchtigen oder unverhältnismäßige Lasten aufzuerlegen. - Verweise auf externe Dokumente oder Richtlinien sind unbesorgt, solange diese leicht zugänglich sind, die bestehenden Vertragskonditionen transparent unterstützen und nicht zu wesentlichen Änderungen im Sinne des Arbeitsvertrages führen.

Kriterien für 'potentiell unzulässig': - Eine Klausel fällt unter 'potentiell unzulässig', wenn sie vage und mehrdeutig bleibt, somit erheblichen Interpretationsspielraum lässt und keine ausreichenden Rahmenbedingungen bietet. Dies gilt insbesondere bei Änderungen hinsichtlich Arbeitsplatz, Aufgaben oder Pflichten des Arbeitnehmers ohne klare und nachvollziehbare Vorsorgemaßnahmen oder Präzisierungen. - Verpflichtungen, die möglicherweise übermäßige oder schwer vorhersehbare Belastungen verursachen, sollten transparent adressieren, wie solche Belastungen gemildert oder kompensiert werden. Fehlen klare, verhältnismäßige Regelungen oder sind Anpassungen unangemessen offen gestaltet, gilt erhöhte Vorsicht. - Klauseln, die dem Arbeitge-

ber einseitige Änderungsrechte verleihen, müssen klar einschränken, wie diese im Lichte des Arbeitnehmerschutzes gewährt werden dürfen, um unverhältnismäßige Entscheidungen zu vermeiden.

Policies

1. Eine Klausel, die vorsieht, dass Überstunden pauschal mit dem Bruttojahresgehalt ohne klare Begrenzung der Anzahl der Überstunden abgegolten sind, ist unzulässig, da sie den Arbeitnehmer unverhältnismäßigen Belastungen aussetzen kann.

2. Eine Klausel, die vom Arbeitnehmer verlangt, seine gesamte Arbeitskraft und sein Wissen ausschließlich in den Dienst des Arbeitgebers zu stellen, ist unzulässig, wenn: 1. der spezifische Arbeitsbereich oder die vertraglich festgelegten Aufgaben des Arbeitnehmers nicht ausreichend detailliert beschrieben sind, wodurch eine unverhältnismäßige Einschränkung der beruflichen Freiheit entsteht, 2. keine zulässigen Ausnahmen für parallele Tätigkeiten unter bestimmten Bedingungen vorgesehen sind, es sei denn, eine solche Exklusivität ist im speziellen Kontext der Tätigkeit notwendig und verhältnismäßig, 3. keine klaren, überprüfbaren und verhältnismäßigen Bedingungen definiert sind, die es dem Arbeitnehmer erlauben, Nebenbeschäftigungen oder Interessen anderer Unternehmen wahrzunehmen, wenn solche nicht im speziellen Kontext der Tätigkeit erforderlich sind.

3. Eine Klausel, die vorsieht, dass der Arbeitnehmer seine Aufgaben ohne ausdrückliche Begrenzung der tätigkeitsspezifischen Verantwortung und des Arbeitsplatzes wahrnehmen muss, ist unzulässig, es sei denn, sie enthält klare, überprüfbare und verhältnismäßige Bedingungen, unter denen Standort- oder Tätigkeitswechsel stattfinden dürfen. Diese Bedingungen müssen sowohl spezifische betriebliche Gründe als auch die Interessen und die Zumutbarkeit für den Arbeitnehmer berücksichtigen.

A.1.7. Garnishment

Component

Du bewertest die Klausel als 'unbedenklich' wenn sie ausdrücklich gängige Standards und Rechtsvorschriften einhält, dem Arbeitnehmer keine wesentlichen Ansprüche entzieht, und klar definierte Rechte und Pflichten ohne versteckte oder ungewöhnliche Bedingungen enthält.

Eine Klausel wird als 'potentiell unzulässig' eingestuft, wenn sie Elemente enthält, die das Gleichgewicht der Rechte zwischen Arbeitgeber und Arbeitnehmer erheblich stören

könnten, auch wenn sie keine klare Verletzung der geltenden Rechtsprechung darstellen. Dazu gehören Klauseln, die restriktiv auf die Ansprüche des Arbeitnehmers einwirken könnten, insbesondere im Zusammenhang mit der Aufrechnung von Forderungen, sowie unklar definierte Bedingungen, die zu einer unterschiedlichen Auslegung führen können. Besonderes Augenmerk liegt dabei auf der Gefahr einer unangemessenen Benachteiligung durch implizite oder explizite Abtretungen ohne gesonderte Zustimmung, auch wenn die unmittelbare Formulierung der Klausel gängige Praktiken reflektiert.

Policies

1. Eine Klausel, die in den AGB ein generelles Pfändungs- oder Abtretungsverbot für die Forderungen des Arbeitnehmers festlegt, ist nach § 308 Nr. 9 lit. a BGB unwirksam.

2. Eine Klausel, die eine Kostenpauschale von mehr als 10 € vorsieht, ist nach unwirksam.

A.1.8. Other

Component

Eine Klausel, die in den AGB ein generelles Pfändungs- oder Abtretungsverbot für die Forderungen des Arbeitnehmers festlegt, ist nach § 308 Nr. 9 lit. a BGB unwirksam. Eine Klausel, die eine Kostenpauschale von mehr als 10 € vorsieht, ist nach unwirksam.

Policies

1. Eine Verpflichtung zur Verschwiegenheit über schon vorhandenene Kenntnisse oder öffentlich bekannte Tatsachen sind unwirksam. Offenkundig sind Tatsachen, wenn sie aufgrund allgemein zugänglicher Quellen erschlossen werden können oder schon allgemein bekannt sind (vgl. § 291 ZPO).

2. Eine Klausel ist nur dann unwirksam, wenn sie das Recht des Arbeitnehmers auf eine Nebenbeschäftigung pauschal und ohne Prüfung der tatsächlichen Beeinträchtigung der Haupttätigkeit oder der berechtigten Interessen des Arbeitgebers einschränkt. Eine Regelung ist jedoch als unbedenklich einzustufen, wenn eine Zustimmungspflicht vor allem darauf abzielt, berechtigte Interessen des Arbeitgebers zu schützen und keine unverhältnismäßige Hürde für den Arbeitnehmer darstellt.

A.1.9. Vacation

Component

Eine Klausel wird als 'unbedenklich' bewertet, wenn sie die gesetzlichen Mindestanforderungen erfüllt und transparent, verständlich und fair die Interessen beider Parteien, insbesondere die der Arbeitnehmer, wahrt. Sie soll keine über die gesetzlichen Bestimmungen hinausgehenden restriktiven Regelungen oder Unklarheiten hinsichtlich der Urlaubsgewährung oder –verfall beinhalten. Dabei sind folgende Aspekte zu berücksichtigen: - Der gesetzliche Urlaubsanspruch muss unmissverständlich als mindestens vier Wochen für eine Fünf-Tage-Woche anerkannt sein. - Jegliche Regelungen für Zusatzurlaub müssen klar und fair deklariert sein, ohne Benachteiligungen bei Krankheit oder Beendigung des Arbeitsverhältnisses. - Die Urlaubsplanung sollte ohne übermäßige Hürden oder Einschränkungen hinsichtlich betrieblicher Bedürfnisse erfolgen.

Eine Klausel gilt als 'potentiell unzulässig', wenn sie implizieren könnte, dass die Arbeitnehmerrechte durch rigide und teils unverhältnismäßige Bedingungen eingeschränkt werden und obwohl sie im lauten Wortlaut nicht vollständig unvereinbar mit dem Gesetz scheinen, durch unklare Formulierungen oder umstrittene Anwendungen erhebliche Einschränkungen verursachen könnten. Besonders kritisch ist: - Eine unklare Trennung oder widersprüchliche Regelungen zwischen gesetzlichem und vertraglichem Urlaub, die Verwirrung oder eine Benachteiligung nahelegen. - Normen, die bei Krankheit, Arbeitspausen oder Beendigung des Arbeitsverhältnisses den Urlaubsanspruch unverhältnismäßig beschränken. - Verfallsregelungen, die über gesetzlich übliche Praktiken hinausgehen und den Verlust von Urlaubsansprüchen involvieren, der nicht angemessen gerechtfertigt ist.

Es ist wichtig, die spezifischen Klauseln im Kontext zu sehen und keine Verstöße gegen allgemein anerkannte Standards in der HR- bzw. Arbeitspraxis zu übersehen, die die Fairness und Klarheit im Allgemeinen einschränken könnten.

Policies

1. Eine Klausel, die den Ausschluss der Vererblichkeit von Urlaubsabgeltungsansprüchen vorsieht, ist gemäß § 1922 BGB i.V.m. § 7 Abs. 4 BUrlG und der Rechtsprechung des Bundesarbeitsgerichts unwirksam und somit unzulässig, da der Urlaubsanspruch auch im Todesfall des Arbeitnehmers auf die Erben übergeht.

2. Eine Klausel, die einen Urlaubsanspruch von weniger als vier Wochen pro Kalenderjahr gemäß dem Bundesurlaubsgesetz vorsieht, ist unzulässig. Für eine

fünf-Tage-Woche müssen dies mindestens 20 Arbeitstage sein, für eine sechs-Tage-Woche mindestens 24 Werktage

3. Eine Klausel, die keine Regelungen für das Übertragen von Urlaub ins nächste Kalenderjahr bis mindestens zum 31. März enthält oder unangemessen kurze Verfallsfristen für Resturlaub setzt, ist potentiell unzulässig, sofern keine ausreichende Begründung für kürzere Fristen gegeben wird

A.1.10. Compensation

Component

Eine Klausel wird als 'unbedenklich' eingestuft, wenn sie klar, präzise und im Einklang mit gesetzlichen Bestimmungen und tariflichen Vereinbarungen steht. Sie lässt keine widersprüchlichen Interpretationen zu und hat keinerlei nachteilige Auswirkungen auf den Arbeitnehmer. Freiwillige Leistungen gelten als unbedenklich, wenn sie ohne zukünftige Verpflichtung deklariert sind und Bedingungen sowie Höhe etwaiger Rückforderungen klar und transparent geregelt sind. Regelungen zu Überstunden müssen explizit festlegen, wie und unter welchen Bedingungen Überstunden vergütet oder durch Freizeit ausgeglichen werden, in Übereinstimmung mit arbeitsrechtlichen Vorschriften. Es muss eindeutig ausgeschlossen sein, dass Arbeitszeitänderungen ohne faire Berücksichtigung von Einwänden erfolgen können. Klauseln über variable Vergütungen oder Zuschläge sind unbedenklich, sofern die Berechnungsweisen eindeutig und nachvollziehbar dargestellt sind und keine Abhängigkeiten von undefinierten oder subjektiven Kriterien bestehen.

Eine Klausel wird als 'potentiell unzulässig' angesehen, wenn ihre Bedingungen unklar oder unpräzise sind und zu verschiedenen Auslegungsmöglichkeiten führen können, wodurch potenziell nachteilige Auswirkungen auf den Arbeitnehmer möglich sind. Dazu zählen Bedingungen für zukünftige Vergütungen, die von unklar definierten oder subjektiven Kriterien abhängen, wie vage formulierte Zielvereinbarungen oder Leistungskriterien. Klauseln, die keine klaren Mechanismen zur Abgeltung von Überstunden beschreiben oder deren Abgeltung von rein subjektiven Genehmigungen abhängig machen, sind ebenfalls potentiell unzulässig. Ebenso betroffen sind Regelungen, die übermäßige Abzüge bei Provisions- oder Bonuszahlungen ohne klar umrissene Rahmenbedingungen enthalten oder Umstände, unter denen Leistungen unwiderruflich verfallen könnten. Besondere Vorsicht ist bei Klauseln geboten, deren Bedingungen zu mehrdeutigen Interpretationen führen oder keine klaren Handlungsanweisungen anbieten, was potenzielle rechtliche Konflikte nach sich ziehen könnte.

Policies

1. Eine Ausschlussfrist, die die Ansprüche des Arbeitnehmers auf Haftung bei Vorsatz, den gesetzlichen Mindestlohn oder unverzichtbare Rechte einschränkt, ist nach unwirksam (BAG Urteil Urteil vom 30. Januar 2019 zum Az.: 5 AZR 43/18).

2. Die pauschale Verschwiegenheitsklausel über Gehalt stellt eine unangemessene Benachteiligung des Arbeitnehmers entgegen den Geboten von Treu und Glauben gemäß § 307 BGB dar und ist damit unwirksam. (LAG Mecklenburg-Vorpommern, Urteil vom 21. Oktober 2009 - 2 Sa 183/09).

3. Eine vertraglich vereinbarte Vergütung, die unter dem gesetzlichen Mindestlohn von 12,41 € pro Arbeitsstunde liegt, verstößt gegen das Mindestlohngesetz und ist damit unwirksam.

4. Eine Klausel, die vorschreibt, betriebliche Schulungen nicht zu entlohnen, ist unwirksam.

5. Eine Klausel, die einen automatischen Pausenabzug vorschreibt, ohne dass tatsächlich eine Pause genommen werden muss, ist unzulässig, da sie gegen die arbeitsrechtlichen Pausenregelungen verstößt.

6. Eine Klausel, die eine pauschale Abgeltung von Überstunden oder ähnlichen Leistungen in einem Betrag vorsieht, ohne klare Unterscheidung der abzugeltenden Stunden oder spezifische Bedingungen für Anpassungen festlegt, ist unzulässig, da sie die Rechte des Arbeitnehmers auf faire Vergütung und transparente Bedingungen einschränkt.

7. Eine Klausel, die eine pauschale Abgeltung von Überstunden bis zu einer bestimmten Anzahl vorsieht, muss klare Konditionen für den Ausgleich darüber hinausgehender Überstunden enthalten und rechtlich zulässig sein. Falls gesetzliche Regelungen zur pauschalen Überstundenabgeltung erfüllt sind, kann die Klausel als unbedenklich gelten.

8. Eine Klausel, die Überstunden pauschal mit dem Gehalt abgilt und über einen bestimmten Umfang hinausgehende Überstunden nur vage mit Ausgleichsfreizeit kompensiert, ist unzulässig, es sei denn, es gibt detaillierte Regelungen zur Handhabung solcher Ausgleiche, um Transparenz sicherzustellen.

9. Eine Klausel, die vorgibt, alle tariflichen Ansprüche pauschal durch andere Leistungen abzugelten, ohne transparente und spezifische Bedingungen oder eine

klare Aufschlüsselung der Abgeltung zu liefern, ist unzulässig, außer wenn ein anerkannter Tarifvertrag existiert, der solche Abgeltungen explizit und klar regelt.

10. Eine Klausel, die die pauschale Abgeltung von Überstunden ohne konkrete Nennung von Ausgleichsmechanismen vorsieht, ist potentiell unzulässig, es sei denn, es existiert ein geregeltes Genehmigungsverfahren durch den Vorgesetzten und die rechtlichen Ansprüche auf Vergütung sind anderweitig gesichert.

11. Eine Regelung, die Außendienstmitarbeitern jegliche Überstundenvergütung verweigert, ist unzulässig, es sei denn, es wird explizit festgelegt, dass die gesamte Arbeitszeit durch ein fair kalkuliertes Gehalt zu einem umfassend rechtlich konformen Niveau vergütet ist und im Einklang mit den arbeitsrechtlichen Schutzgesetzen steht.

12. Eine Vergütungsklausel, die keine genaue Aussage zur Einhaltung des gesetzlichen Mindestlohns oder zur Anzahl der Arbeitsstunden macht, ist unzulässig, da sie die Möglichkeit schafft, gesetzlich vorgeschriebene Mindestlohn- und Arbeitszeitstandards zu unterschreiten, es sei denn, es sind konkreter Nachweis und Klarheit der Abdeckung dieser Mindestanforderungen gegeben.

A.1.11. Statute of Limitations

Component

Eine Klausel wird als 'unbedenklich' eingestuft, wenn sie:
- **Eindeutigkeit und Klarheit**: Eindeutig formuliert ist und klare, konkret benannte Fristen enthält, die den Parteien die Bedingungen verständlich und nachvollziehbar erläutern. Beginn und Umfang der Fristen müssen definiert sein, sodass keine Unklarheiten bestehen, die das Verständnis erschweren.
- **Beachtung rechtlicher Standards**: Die gesetzlichen Vorgaben für Verjährungsfristen werden nicht verkürzt, es sei denn, diese sind rechtlich zulässig und transparent begründet. Dabei dürfen keinerlei unangemessene Benachteiligungen entstehen. Klauseln sollten notwendige gesetzliche oder tarifliche Grundlagen für Anpassungen nennen.
- **Aktuelle rechtliche Rahmenbedingungen**: Die Klausel erfüllt die Anforderungen an Fristenformen (z.B. Textform), die rechtlich anerkannt sind. Änderungen, die im rechtlichen Umfeld wie die Unzulässigkeit der reinen Schriftform für erste Geltendmachungen, berücksichtigt werden müssen.
- **Klare und umfassende Ausnahmeregelungen**: Ausnahmen wie für Vorsatz, grobe Fahrlässigkeit, und Schutzstandards (z.B. gesetzlicher Mindestlohn) sind festgehalten, um Anforderungen an die Unverzichtbarkeit einzuhalten.

Eine Klausel wird als 'potentiell unzulässig' eingestuft, wenn:
- **Mangel an Spezifikationen**: Unklare oder nicht spezifizierte Fristen bestehen, die Verwirrung schaffen und die Nachvollziehbarkeit der Erfüllung erschweren. Ausdrückliche Nennung von Anwendungszeitpunkten und Bezugsperioden für Fristen fehlt oder widerspricht üblichen Standards.
- **Kritische Fristgestaltung ohne Kontext**: Fristen, die auf den ersten Blick nicht unzulässig erscheinen, jedoch abweichen von der Branchenpraxis oder ohne hinreichende Begründung durch rechtliche oder vertragliche Notwendigkeiten stehen.
- **Inhaltliche und strukturelle Inkohärenz**: Inhaltliche Vermischungen in Klauseln führen zu Benachteiligungen, weil sie Regelungsgebiete überschneiden, ohne klare Abgrenzung von Fristenbereichen. Insbesondere eine schwer nachvollziehbare Struktur oder das Fehlen einer kohärenten Regelung kann zu einer Beeinträchtigung der Fairness führen.

Es ist sicherzustellen, dass die Klausel nur bei erheblichen Unklarheiten, die die Gerechtigkeit unverhältnismäßig beeinträchtigen könnten, als 'potentiell unzulässig' klassifiziert wird.

Policies

1. Eine Klausel, die eine Ausschlussfrist von weniger als drei Monaten für die Geltendmachung von Ansprüchen vorsieht, ist unzulässig, es sei denn, es gibt spezifische gesetzliche oder tarifvertragliche Bestimmungen, die solche kurzen Fristen rechtfertigen.

2. Eine Klausel, die verlangt, dass Ansprüche in Schriftform (statt mindestens in Textform) geltend gemacht werden müssen, ist unzulässig, da sie Arbeitnehmer unangemessen benachteiligt.

3. Eine Klausel, die eine doppelte Ausschlussfrist festlegt, muss eine Ausnahme für unverzichtbare Ansprüche wie Mindestlohn, Ansprüche wegen Verletzung des Lebens, des Körpers oder der Gesundheit sowie Haftungsansprüche wegen vorsätzlicher Pflichtverletzung enthalten, ansonsten ist sie unzulässig

4. Eine Klausel, die eine doppelte Ausschlussfrist ohne angemessene Ausnahmeregelung für unverzichtbare Ansprüche wie Mindestlohn, Ansprüche wegen Verletzung des Lebens, des Körpers oder der Gesundheit sowie Haftungsansprüche wegen vorsätzlicher Pflichtverletzung vorsieht, ist unzulässig.

5. Klauseln, die den Verfall von Ansprüchen in Verbindung mit einer Kündigungsschutzklage vorsehen, müssen eine angemessene Frist für die Geltendmachung nach rechtskräftiger Entscheidung gewähren. Eine Frist von drei Monaten gilt als

angemessen und ist unbedenklich, sofern keine weiteren unzumutbaren Bedingungen auferlegt werden.

6. Eine Klausel, die sowohl schriftliche als auch textliche Benachrichtigungsformen verlangt, ist potentiell unzulässig, es sei denn, die schriftliche Form wird nur als eine von mehreren gleichwertigen Optionen angeboten.

7. Verkürzungen der regelmäßigen Verjährungsfrist auf weniger als drei Jahre sind unzulässig, es sei denn, die Klausel enthält spezifische gesetzliche oder tarifliche Ausnahmen oder der Vertrag enthält eine separierte Regelung, die gesetzlich geregelte Fristen ausdrücklich ausnimmt.

8. Bei der Bewertung der Zulässigkeit einer Verjährungsklausel ist zu berücksichtigen, ob der Vertrag an anderer Stelle spezifische Ausnahmeregelungen enthält, die gesetzliche Vorgaben erfüllen und unverzichtbare Ansprüche absichern.

9. Eine Klausel, die eine Ausschlussfrist von genau drei Monaten für die Geltendmachung von Ansprüchen vorsieht, ist grundsätzlich als unbedenklich zu betrachten, es sei denn, sie enthält keine Ausnahmen für unverzichtbare Ansprüche oder widerspricht tarifvertraglichen Bestimmungen.

10. Die Prüfung sollte berücksichtigen, ob die Frist in spezifischen Kontexten gesetzlich gerechtfertigt ist.

11. Eine Ausschlussfrist, die in Textform zu erfolgen hat, ist grundsätzlich unzulässig, wenn sie kürzer als drei Monate für die gerichtliche Geltendmachung ist, es sei denn, eine solche Verkürzung wird durch spezifische gesetzliche oder tarifliche Bestimmungen gerechtfertigt.

A.1.12. Penalty Clause

Component

Eine Klausel ist als 'unbedenklich' einzustufen, wenn sie die legitimen Interessen des Arbeitgebers wahrt, ohne die grundlegenden Rechte der Arbeitnehmer unverhältnismäßig zu beschränken. Besonders bei Klauseln zu Nebenbeschäftigungen muss eine Zustimmungspflicht des Arbeitgebers klar definieren, unter welchen Umständen die Zustimmung verweigert werden kann. Dies muss transparent und begründet erfolgen, wobei primär die Beeinträchtigung der Haupttätigkeit und genau spezifizierte Interessen des Arbeitgebers im Fokus stehen sollten. Die Verweigerung der Zustimmung

oder ihr Widerruf müssen sachlich gerechtfertigt sein und mit angemessenen Ausgleichsmechanismen verbunden werden, um sicherzustellen, dass die Arbeitnehmerrechte nicht unverhältnismäßig eingeschränkt werden.

Eine Klausel gilt als 'potentiell unzulässig', wenn sie Elemente enthält, die möglicherweise die Rechte der Arbeitnehmer unangemessen einschränken, auch wenn es Hinweise gibt, dass die Einschränkung durch Praxis oder rechtliche Regelungen gerechtfertigt sein könnte. Bei der Einstufung ist eine genaue Betrachtung der Formulierungen notwendig, insbesondere bei vagen oder unpräzisen Bedingungen, die eine potenzielle Unzulässigkeit andeuten. Auch restriktive Klauseln, die auf dem Papier problematisch erscheinen, könnten aufgrund von Verhältnismäßigkeit oder bestehenden Ausgleichsmechanismen akzeptabel sein. In solchen Fällen müssen die Bedingungen klar und verhältnismäßig beschrieben sein, und es sollte eine transparente Abwägung von Interessen gegeben sein.

A.1.13. Overtime

Component

Eine Klausel ist als 'unbedenklich' einzustufen, wenn sie den gesetzlichen Standards entspricht und den Schutz des Arbeitnehmers durch konkrete Maßnahmen berücksichtigt. Die Klausel muss klar und verständlich sein, mit eindeutigen Begrifflichkeiten, und Überstunden dürfen nur im gesetzlichen Rahmen angeordnet werden. Eine ausdrückliche Anordnung durch Vorgesetzte oder eine vorab vereinbarte Regelung, die betriebliche Notwendigkeiten berücksichtigt, ist erforderlich. Zudem müssen Arbeitszeiten im Einklang mit den gesetzlichen Vorgaben limitiert werden und die zumutbare Arbeitsbelastung betont werden.

Eine Klausel wird als 'potentiell unzulässig' betrachtet, wenn sie vage Formulierungen enthält, keine klaren gesetzlichen Begrenzungen vorgibt oder Mechanismen zur Entlohnung von Überstunden fehlen. Dies gilt insbesondere, wenn die Klausel keine klaren Bedingungen für die Anordnung von Überstunden festlegt oder der Arbeitgeber einseitig Rechte erhält, ohne nachvollziehbare betriebliche Notwendigkeiten oder gesetzliche Regularien zu beachten. Die Abwesenheit der Zustimmung des Arbeitnehmers oder die Möglichkeit, die Klausel missbräuchlich auszulegen, kann eine unangemessene Benachteiligung des Arbeitnehmers darstellen und erfordert eine genauere Prüfung.

Policies

1. Klauseln zur Abgeltung von Überstunden sind unwirksam, wenn sie festlegen, dass Überstunden pauschal mit dem Gehalt abgegolten sind oder wenn sie nicht transparent festlegen, wie viele Überstunden ohne zusätzliche Vergütung zu

leisten sind, und dabei keine unzulässigen Rechtsfolgen wie etwa die Unterschreitung des Mindestlohns oder ein auffälliges Missverhältnis zwischen Leistung und Gegenleistung eintreten.

2. Umfang von über 8 Stunden Überstunden pro Woche ist nur dann unzulässig, wenn der Arbeitsvertrag eine reguläre Arbeitszeit von 40 Stunden pro Woche ohne Ausnahmegenehmigung festlegt. Zudem muss berücksichtigt werden, ob die Klausel allgemein geltendem Arbeitsrecht entspricht und ob sie im Einzelfall (z.B. Branchenstandard oder individuelle Vereinbarungen) sich im gesetzlichen Rahmen bewegt.

3. Eine Klausel, die Überstunden vorschreibt, ist nicht unzulässig, wenn sie sich auf die Einhaltung geltender gesetzlicher Bestimmungen beruft und die genannte Stundenzahl durch das Arbeitszeitgesetz abgedeckt ist, solange die Arbeitsbedingungen im Vertrag nicht explizit gegen diese Bestimmungen verstoßen.

Policies

1. Ein vollständiges Wettbewerbsverbot ohne Ausnahmen für rein finanzielle Minderheitsbeteiligungen bis zu 5 % der Gesellschaftsanteile ist problematisch.

2. Ein nachvertragliches Wettbewerbsverbot ohne Karenzentschädigung ist unwirksam.

3. Ein nachvertragliches Wettbewerbsverbot von mehr als zwei Jahren ist unwirksam.

4. Ein nachvertragliches Wettbewerbsverbot mit einer Karenzentschädigung, die weniger als 50 % des zuletzt bezogenen Bruttomonatsgehalts des Arbeitnehmers beträgt, inklusive aller festen und variablen Gehaltsbestandteile (z. B. Prämien, Boni), ist unwirksam.

5. Eine Klausel, die die Verschwiegenheitspflicht des Arbeitnehmers bezüglich der Vertragsinhalte und des Gehalts regelt, ist unzulässig, wenn sie bekannte Tatsachen nicht ausnimmt oder die Arbeitnehmer unverhältnismäßig benachteiligt, da Informationen über Gehälter zunehmend als Teil der Arbeitnehmerrechte auf Transparenz betrachtet werden können.

6. Ein unbeschränkter Widerruf der Zustimmung zu einer Nebenbeschäftigung seitens des Arbeitgebers ohne sachlich gerechtfertigten Grund ist unwirksam und daher unzulässig.

7. Eine Klausel, die das Recht des Arbeitnehmers auf eine Nebenbeschäftigung einschränkt, ist unwirksam, wenn sie pauschal verbietet, unverhältnismäßig strenge Bedingungen setzt oder über die berechtigten Interessen des Arbeitgebers hinausgeht, ohne das Arbeitszeitgesetz, Konkurrenzverbot oder die Beeinträchtigung der Haupttätigkeit angemessen zu berücksichtigen. Eine pauschale Zustimmungspflicht ist unzulässig, es sei denn, sie geht mit einer widerspruchsfähigen Begründungspflicht für die Ablehnung der Zustimmung einher.

8. Eine Klausel, die die Verschwiegenheitspflicht auf bereits vorhandene Kenntnisse des Arbeitnehmers ausweitet und diese als Betriebsgeheimnisse behandelt, ist unzulässig, da sie über die berechtigten Interessen des Arbeitgebers hinausgeht und bestehende Arbeitnehmerrechte unverhältnismäßig einschränkt.

A.2. Extended Evaluation Metrics

This sections presents detailed performance metrics for the four experimental conditions: $(APO, GPT-4o)$, $(Baseline, GPT-4o)$, $(APO, GPT-4o-mini)$, and $(Baseline, GPT-4o-mini)$, expanding on the results discussed in Chapter 4. For each condition, we report category-specific overall accuracy, F1-score, precision, and recall, as well as label-level F1-score, precision, and recall.

Configuration (APO, GPT-4o)													
Category	Overall				Fair			Unfair			Void		
	Accuracy	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision
Other	0.69	0.56	0.60	0.70	0.81	0.70	0.96	0.21	0.60	0.13	0.67	0.50	1.00
Compensation	0.72	0.64	0.75	0.68	0.80	0.68	0.97	0.27	0.75	0.17	0.86	0.82	0.90
Penalty Clause	0.50	0.51	0.63	0.60	0.57	0.40	1.00	0.30	0.79	0.18	0.67	0.71	0.62
Execution	0.50	0.44	0.68	0.57	0.62	0.45	0.98	0.27	0.91	0.16	0.00	0.00	0.00
Termination	0.68	0.44	0.54	0.43	0.83	0.74	0.94	0.07	0.12	0.05	0.44	0.75	0.30
Tasks	0.49	0.46	0.71	0.51	0.62	0.46	0.97	0.09	0.67	0.05	0.67	1.00	0.50
Illness	0.80	0.67	0.76	0.66	0.89	0.81	1.00	0.30	0.60	0.20	0.82	0.88	0.78
Vacation	0.59	0.60	0.75	0.60	0.68	0.54	0.93	0.38	0.70	0.26	0.75	1.00	0.60
Statute of Limitations	0.62	0.56	0.62	0.60	0.58	0.44	0.88	0.40	0.50	0.33	0.71	0.92	0.58
Format	0.58	0.57	0.63	0.64	0.62	0.45	1.00	0.43	0.43	0.43	0.67	1.00	0.50
Garnishment	0.85	0.72	0.88	0.75	0.77	0.62	1.00	0.40	1.00	0.25	1.00	1.00	1.00
Overtime	0.62	0.69	0.83	0.75	0.67	0.50	1.00	0.40	1.00	0.25	1.00	1.00	1.00
Cut-off Periods	0.89	0.80	0.94	0.75	0.00	0.00	0.00	0.67	1.00	0.50	0.93	0.88	1.00
Overall	0.63	0.59	0.7	0.61	0.73	0.59	0.96	0.26	0.65	0.16	0.77	0.85	0.70

Figure A.1.: Performance metrics for APO-optimized prompts using GPT-4o.

Configuration (Baseline, GPT-4o)													
Category	Overall				Fair			Unfair			Void		
	Accuracy	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision
Other	0.74	0.63	0.78	0.59	0.84	0.75	0.96	0.25	0.60	0.16	0.80	1.00	0.67
Compensation	0.69	0.55	0.56	0.58	0.79	0.73	0.86	0.12	0.25	0.08	0.75	0.71	0.80
Penalty Clause	0.63	0.56	0.60	0.62	0.75	0.67	0.85	0.34	0.64	0.23	0.59	0.48	0.77
Execution	0.75	0.61	0.74	0.61	0.84	0.75	0.96	0.37	0.73	0.25	0.00	0.00	0.00
Termination	0.67	0.36	0.45	0.36	0,79	0,71	0.89	0.29	0.62	0.19	0.00	0.00	0.00
Tasks	0.68	0.55	0.71	0.57	0.80	0.67	0.98	0.13	0.67	0.07	0.73	0.80	0.67
Illness	0.74	0.57	0.69	0.56	0.86	0.77	0.98	0.38	0.8	0.25	0.47	0.50	0.44
Vacation	0.80	0.62	0.57	0.77	0.89	0.89	0.89	0.45	0.50	0.42	0.50	0.33	1.00
Statute of Limitations	0.53	0.41	0.39	0.45	0.59	0.50	0.73	0.00	0.00	0.00	0.64	0,67	0,62
Format	0.54	0.40	0,45	0.42	0.65	0.91	0.50	0.55	0.43	0.75	0.00	0.00	0..0
Garnishment	0.65	0.53	0.47	0.62	0.88	0.88	0.88	0.00	0.00	0.00	0.71	0.55	1.00
Overtime	0.50	0.52	0.78	0.58	0.50	0.33	1.00	0.40	1.00	0.25	0.67	1.00	0.50
Cut-off Periods	0.78	0.44	0.44	0.44	0.00	0.00	0.00	0.00	0.00	0.00	0.88	0.88	0.88
Overall	0.70	0.58	0.62	0.61	0.81	0.73	0.90	0.28	0.56	0.19	0.65	0.58	0.73

Figure A.2.: Performance metrics for the baseline prompt using GPT-4o

Configuration (APO, GPT-4o-mini)													
Category	Overall				Fair			Unfair			Void		
	Accuracy	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision
Other	0.47	0.4	0.48	0.61	0.61	0.44	0.98	0.2	0.9	0.11	0.4	0.5	0.33
Compensation	0.57	0.55	0.63	0.74	0.63	0.47	0.95	0.26	1.0	0.15	0.76	0.74	0.78
Penalty Clause	0.3	0.33	0.53	0.49	0.29	0.17	0.93	0.26	0.93	0.15	0.43	0.38	0.5
Execution	0.48	0.43	0.57	0.67	0.6	0.43	0.98	0.26	0.91	0.15	0.0	0.0	0.0
Termination	0.33	0.36	0.47	0.55	0.43	0.28	0.96	0.14	0.62	0.08	0.5	0.75	0.38
Tasks	0.29	0.43	0.59	0.74	0.35	0.22	1.0	0.09	1.0	0.05	0.83	1.0	0.71
Illness	0.37	0.39	0.62	0.56	0.48	0.32	1.0	0.19	1.0	0.1	0.5	0.38	0.75
Vacation	0.42	0.38	0.49	0.52	0.49	0.33	1.0	0.36	0.9	0.22	0.29	0.33	0.25
Statute of Limitations	0.57	0.42	0.53	0.43	0.55	0.38	1.0	0.0	0.0	0.0	0.71	0.92	0.58
Format	0.5	0.46	0.66	0.6	0.17	0.09	1.0	0.63	0.71	0.56	0.6	1.0	0.43
Garnishment	0.7	0.55	0.71	0.75	0.4	0.25	1.0	0.25	1.0	0.14	1.0	1.0	1.0
Overtime	0.88	0.86	0.83	0.94	0.91	0.83	1.0	0.67	1.0	0.5	1.0	1.0	1.0
Cut-off Periods	0.78	0.44	0.44	0.44	0.0	0.0	0.0	0.0	0.0	0.0	0.88	0.88	0.88
Overall	0.44	0.47	0.58	0.64	0.51	0.34	0.98	0.23	0.85	0.13	0.68	0.71	0.65

Figure A.3.: Performance metrics for APO-optimized prompts using GPT-4o-mini.

Configuration (Baseline, GPT-4o-mini)													
Category	Overall				Fair			Unfair			Void		
	Accuracy	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision	F1	Recall	Precision
Other	0.7	0.34	0.36	0.41	0.82	0.72	0.95	0.19	0.5	0.12	0.0	0.0	0.0
Compensation	0.65	0.51	0.6	0.53	0.8	0.74	0.86	0.14	0.38	0.09	0.6	0.47	0.84
Penalty Clause	0.51	0.48	0.56	0.58	0.62	0.49	0.87	0.33	0.86	0.21	0.47	0.38	0.62
Execution	0.81	0.56	0.55	0.57	0.89	0.87	0.91	0.22	0.27	0.19	0.0	0.0	0.0
Termination	0.65	0.32	0.33	0.36	0.78	0.71	0.86	0.19	0.38	0.13	0.0	0.0	0.0
Tasks	0.69	0.5	0.54	0.55	0.82	0.71	0.97	0.07	0.33	0.04	0.6	0.6	0.6
Illness	0.61	0.42	0.46	0.51	0.79	0.67	0.97	0.21	0.6	0.12	0.27	0.25	0.29
Vacation	0.66	0.37	0.35	0.39	0.78	0.76	0.8	0.32	0.4	0.27	0.0	0.0	0.0
Statute of Limitations	0.33	0.31	0.42	0.38	0.52	0.38	0.86	0.12	0.5	0.07	0.29	0.25	0.33
Format	0.46	0.21	0.15	0.33	0.63	1.0	0.46	0.0	0.0	0.0	0.0	0.0	0.0
Garnishment	0.4	0.31	0.39	0.27	0.33	0.25	0.5	0.0	0.0	0.0	0.6	0.55	0.67
Overtime	0.62	0.4	0.42	0.56	0.8	0.67	1.0	0.4	1.0	0.25	0.0	0.0	0.0
Cut-off Periods	0.56	0.36	0.42	0.31	0.0	0.0	0.0	0.0	0.0	0.0	0.71	0.62	0.83
Overall	0.64	0.48	0.54	0.51	0.78	0.7	0.88	0.21	0.44	0.14	0.46	0.37	0.6

Figure A.4.: Performance metrics for the baseline prompt using GPT-4o-mini.

B. Additional Results of Experiment II

B.1. Category-Specific CoT Prompts

This section presents the category-specific CoT prompts created using the prompt design methodology outlined in section 5.

B.1.1. Cut-off Periods

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder
'unbedenklich', 'potentiell unzulssig' oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt.
   Das ist der Fall, wenn:
   - Die Frist fr die auergerichtliche Geltendmachung krzer als 3 Monate
       ist und/oder
     die Frist fr die anschlieende gerichtliche Geltendmachung ebenfalls
        krzer als 3 Monate ist
     (z. B. 2 + 2 Monate, 3 + 2 Monate usw.).
   - Ein strengeres Formerfordernis als die in der Rechtsprechung
       anerkannte Textform verlangt wird
     (z. B. ausschlielich Schriftform mit eigenhndiger Unterschrift) und
        damit die Textform ausgeschlossen ist.
   - Bestimmte zwingend unverfallbare Ansprche (z. B. nach dem MiLoG, bei
       Vorsatz, Verletzung von Leben, Krper
     oder Gesundheit) nicht ausgenommen werden.
   - Im Vertrag eine zustzliche (unzulssige) Verkrzung der gesetzlichen
       Verjhrungsfrist vereinbart wird
     (z. B. nur 1 Jahr fr alle Ansprche).
```

```
2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist
   und mglicherweise gegen arbeitsrechtliche Vorgaben verstt. Das ist
       insbesondere der Fall, wenn:
   - Die Klausel zwar eine zweistufige Frist (z. B. 3 + 3 Monate) vorsieht
       , aber ein zu strenges
     oder unzutreffendes Formerfordernis (z. B. nur Schriftform, keine
         Textform) enthlt.
   - Unklar bleibt, ob zwingend unverfallbare Ansprche (z. B. Mindestlohn,
        vorstzliche Haftung)
     tatschlich ausgenommen sind oder die Formulierung widersprchlich ist.
   - Die Fristen zwar formal 3 + 3 Monate betragen, aber weitere
       Bedingungen
     (z. B. Verkrzung bei Nichtreaktion binnen einer sehr kurzen Frist) zu
          rechtlichen Unsicherheiten fhren.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.2. Execution

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich' oder 'potentiell
```

```
    unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist der Fall, wenn:
   - Die zugewiesenen Ttigkeiten sind nicht hinreichend konkretisiert.
   - Vom Arbeitnehmer werden unzulssige Ausknfte zu persnlichen Umstnden
       verlangt.
   - Die Bereitstellung eines geeigneten Arbeitsplatzes ist nicht
       ausreichend garantiert.
   - Der Arbeitgeber hat eine umfassende, jederzeitige
       Versetzungsmglichkeit ohne ausreichende Einschrnkungen.
   - Eine Befristung ist unklar oder fehlerhaft geregelt.
   - Das Recht auf auerordentliche Kndigung wird ausgeschlossen oder
       eingeschrnkt.
   - Die Datenverarbeitung und -weitergabe ist nicht transparent oder
       ausreichend geregelt.
   - Spesen- oder Reisekostenregelungen werden pauschal ausgeschlossen,
       ohne die Entfernung zum Arbeitsort zu bercksichtigen.

2. Falls keine dieser Regeln verletzt wird, muss die Klausel immer als '
    unbedenklich' bewertet werden.

3. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

4. Antworte nur mit einer der beiden Optionen, ohne weitere Erklrungen
    oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.3. Format

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Sie zwingend die Schriftform (mit eigenhndiger Unterschrift) fr
       Vertragsnderungen, Ergnzungen oder Nebenabreden vorschreibt und
       dadurch rechtssicherere oder vereinfachte Formvorschriften (z. B.
       Textform) ausschliet.
   - Sie eine sogenannte doppelte Schriftformklausel enthlt (Dies gilt
       auch fr die nderung des Schriftformerfordernisses) und gleichzeitig
        die gesetzlich anerkannte Textform erschwert oder unmglich macht.
   - Sie mndliche oder formfreie Individualabreden generell fr unwirksam
       erklrt, obwohl diese rechtlich mglich sein knnen.
   - Sie ohne hinreichende Begrndung jede Form von betrieblicher bung
       kategorisch ausschlieen will, obwohl das Gesetz und die
       Rechtsprechung grundstzlich andere Formen des Zustandekommens von
       Vertragsnderungen zulassen.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Der Vertrag zwar ein Schriftformerfordernis vorsieht, aber zugleich
       Ausnahmen oder nachtrgliche formlos mgliche Abreden erwhnt, die
       unklar oder widersprchlich formuliert sind (z. B. Ausnahmsweise
       knnen nderungen auch mndlich vereinbart werden, ohne klare Grenzen).

   - Unklar bleibt, ob zumindest die Textform (z. B. E-Mail) fr
       Vertragsnderungen ausreichend sein soll oder ob rechtlich
       notwendige Formerleichterungen faktisch unterlaufen werden.
   - Die Klausel zwar die Schriftform verlangt, aber nur andeutet, dass
       diese Anforderung in bestimmten Fllen nicht gilt, ohne die
       Voraussetzungen dafr hinreichend transparent zu machen.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.
```

```
4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.4. Illness

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Sie zwingend die Schriftform (mit eigenhndiger Unterschrift) fr
       Vertragsnderungen, Ergnzungen oder Nebenabreden vorschreibt und
       dadurch rechtssicherere oder vereinfachte Formvorschriften (z. B.
       Textform) ausschliet.
   - Sie eine sogenannte doppelte Schriftformklausel enthlt (Dies gilt
       auch fr die nderung des Schriftformerfordernisses) und gleichzeitig
        die gesetzlich anerkannte Textform erschwert oder unmglich macht.
   - Sie mndliche oder formfreie Individualabreden generell fr unwirksam
       erklrt, obwohl diese rechtlich mglich sein knnen.
   - Sie ohne hinreichende Begrndung jede Form von betrieblicher bung
       kategorisch ausschlieen will, obwohl das Gesetz und die
       Rechtsprechung grundstzlich andere Formen des Zustandekommens von
       Vertragsnderungen zulassen.
```

```
2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Der Vertrag zwar ein Schriftformerfordernis vorsieht, aber zugleich
       Ausnahmen oder nachtrgliche formlos mgliche Abreden erwhnt, die
       unklar oder widersprchlich formuliert sind (z. B. Ausnahmsweise
       knnen nderungen auch mndlich vereinbart werden, ohne klare Grenzen).

   - Unklar bleibt, ob zumindest die Textform (z. B. E-Mail) fr
       Vertragsnderungen ausreichend sein soll oder ob rechtlich
       notwendige Formerleichterungen faktisch unterlaufen werden.
   - Die Klausel zwar die Schriftform verlangt, aber nur andeutet, dass
       diese Anforderung in bestimmten Fllen nicht gilt, ohne die
       Voraussetzungen dafr hinreichend transparent zu machen.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.5. Termination

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
```

```
    oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
   arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Sie zwingend die Schriftform (mit eigenhndiger Unterschrift) fr
     Vertragsnderungen, Ergnzungen oder Nebenabreden vorschreibt und
     dadurch rechtssicherere oder vereinfachte Formvorschriften (z. B.
     Textform) ausschliet.
   - Sie eine sogenannte doppelte Schriftformklausel enthlt (Dies gilt
     auch fr die nderung des Schriftformerfordernisses) und gleichzeitig
      die gesetzlich anerkannte Textform erschwert oder unmglich macht.
   - Sie mndliche oder formfreie Individualabreden generell fr unwirksam
     erklrt, obwohl diese rechtlich mglich sein knnen.
   - Sie ohne hinreichende Begrndung jede Form von betrieblicher bung
     kategorisch ausschlieen will, obwohl das Gesetz und die
     Rechtsprechung grundstzlich andere Formen des Zustandekommens von
     Vertragsnderungen zulassen.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
   unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
   Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Der Vertrag zwar ein Schriftformerfordernis vorsieht, aber zugleich
     Ausnahmen oder nachtrgliche formlos mgliche Abreden erwhnt, die
     unklar oder widersprchlich formuliert sind (z. B. Ausnahmsweise
     knnen nderungen auch mndlich vereinbart werden, ohne klare Grenzen).

   - Unklar bleibt, ob zumindest die Textform (z. B. E-Mail) fr
     Vertragsnderungen ausreichend sein soll oder ob rechtlich
     notwendige Formerleichterungen faktisch unterlaufen werden.
   - Die Klausel zwar die Schriftform verlangt, aber nur andeutet, dass
     diese Anforderung in bestimmten Fllen nicht gilt, ohne die
     Voraussetzungen dafr hinreichend transparent zu machen.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
   unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
   vorliegt, entscheide dich fr 'unbedenklich'.
```

```
5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.6. Tasks

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Die gesamte Arbeitskraft des Arbeitnehmers ohne jede Einschrnkung
       oder ohne jeglichen sachlichen Bezug zum konkret geschuldeten
       Aufgabenbereich gefordert wird (z. B. unbedingtes Verbot jeglicher
       Nebenbeschftigung ohne Beachtung der Interessenabwgung oder ohne
       Ausnahmemglichkeit).
   - Nebenbeschftigungen generell oder ohne sachliche Grnde vollstndig
       untersagt werden, ohne einen angemessenen Spielraum fr den
       Arbeitnehmer vorzusehen.
   - Der Arbeitnehmer zur Befolgung smtlicher Weisungen verpflichtet wird,
        ohne dass diese Weisungen in sachlichem Zusammenhang mit der
       arbeitsvertraglich geschuldeten Ttigkeit stehen (etwa eine
       vollstndige Unterwerfung unter alle Anordnungen der Geschftsfhrung,
        ohne Begrenzung durch arbeitsschutz- oder arbeitszeitrechtliche
       Vorgaben).
   - berstunden oder Mehrarbeit jeglicher Art pauschal mit dem Grundgehalt
        abgegolten werden, ohne eine angemessene Begrenzung oder ohne
```

```
      Transparenz, welche Arbeitszeiten konkret erfasst sind.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Die Ttigkeitsbeschreibung oder der Aufgabenbereich zwar im Vertrag
       oder in einer Stellenbeschreibung definiert ist, die
       Arbeitgeberseite sich jedoch ein unbeschrnktes, einseitiges
       nderungsrecht vorbehlt (z. B. nach billigem Ermessen jederzeit
       abweichend regelbar), ohne transparente Grenzen oder Rcksichtnahme
       auf die Interessen des Arbeitnehmers.
   - Eine Befrderung oder Gehaltsentwicklung an unspezifische Kriterien (z
       . B. gute Entwicklung) gekoppelt wird, ohne klare, berprfbare oder
       objektive Mastbe.
   - Die Lage und Verteilung der Arbeitszeit (z. B. welche Wochentage oder
        Schichten) vom Arbeitgeber jederzeit frei einseitig nderbar sein
       soll, ohne Rcksicht auf schutzwrdige Belange des Arbeitnehmers.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.7. Garnishment

```
### Aufgabe ###
```

```
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Ein generelles Verbot der Abtretung und/oder Verpfndung von
       Vergtungsansprchen vorgesehen ist ( 308 Nr. 9 lit. a BGB).
   - Eine Abtretung oder Verpfndung nur mit Zustimmung des Arbeitgebers
       erlaubt ist, ohne dass sachliche Grnde fr eine Verweigerung
       vorgesehen sind oder diese Erlaubnis faktisch unmglich gemacht wird.

   - berzogene oder pauschale Kosten dem Arbeitnehmer fr eine Pfndung,
       Abtretung oder Verpfndung auferlegt werden, die nicht zumindest
       durch eine realistische Schadensberechnung oder eine moderate
       Pauschale begrndet sind.
   - Die Klausel den unpfndbaren Teil des Arbeitsentgelts in unzulssiger
       Weise erfasst oder schmlert.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Die Klausel Aufrechnungsmglichkeiten einseitig regelt, ohne
       klarzustellen, dass lediglich unbestrittene oder rechtskrftig
       festgestellte Forderungen gegeneinander aufgerechnet werden drfen
       und der pfndungsfreie Betrag nicht berhrt wird.
   - Eine Kostenregelung fr Pfndung, Abtretung oder Verpfndung zwar
       besteht, aber unklar bleibt, ob sie im Einzelfall angemessen und an
        den tatschlichen Aufwand des Arbeitgebers angepasst ist.
   - Aus dem Wortlaut nicht eindeutig hervorgeht, ob der gesetzlich
       garantierte Pfndungsschutz (z. B. unpfndbarer Teil des
       Arbeitsentgelts) vollstndig gewahrt bleibt.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.
```

```
5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.8. Other

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Sie unverhltnismig in die Persnlichkeitsrechte des Arbeitnehmers
       eingreift (z. B. verpflichtende medizinische Untersuchungen ohne
       sachlichen Grund, umfassende Verhaltens- oder Wettbewerbsverbote
       ohne berechtigtes Interesse).
   - Eine mitteilungspflichtige nderung persnlicher Verhltnisse ber das
       gesetzlich oder vertraglich Erforderliche hinausgeht (z. B.
       unverzgliche Schwangerschaftsmeldepflicht).
   - Dritte Personen (z. B. Familienangehrige) in einer Weise einbezogen
       werden, die den Arbeitnehmer unverhltnismig in seinen privaten
       Sphrenrechten beschrnkt oder haftbar macht.
   - Die Klausel ein generelles Aufrechnungsverbot ausspricht, obwohl dem
       Arbeitnehmer nach geltendem Recht ein Aufrechnungsrecht bei
       unbestrittenen oder rechtskrftig festgestellten Forderungen zusteht.

   - Vereinbarungen zur Abtretung von Ansprchen oder Nutzungsrechten dem
       Arbeitnehmer wesentliche gesetzlich garantierte Rechte entziehen (z.
```

```
       B. pauschale Abtretung von Schadensersatzansprchen oder
      unbeschrnkte bertragung von Urheber- oder Persnlichkeitsrechten
      ohne angemessene Regelung).
   - Eine Pflicht zur Vorlage eines Fhrungszeugnisses, rztlichen Attests
      oder einer Zuverlssigkeitsprfung ohne objektiv sachlichen Anlass
      oder gesetzliche Grundlage begrndet wird.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Sie eine verzgerte oder teilweise unklare Regelung zu
      Schadensersatzansprchen, Herausgabepflichten oder
      Mitteilungspflichten enthlt, die nicht klarstellt, in welchen Fllen
       und zu welchem Umfang dies erforderlich ist.
   - Datenschutzrechtliche Fragen (z. B. bezglich Foto- und Videonutzung)
      zwar geregelt werden, aber unklar bleibt, ob die Zustimmung frei
      widerruflich ist und die Verarbeitung den gesetzlichen Bestimmungen
       (DSGVO) entspricht.
   - Die Klausel zwar die Offenlegung gewisser persnlicher Daten verlangt
      (z. B. Meldung von Adressnderungen, Behinderungen), die
      Formulierungen aber nicht przise genug abgrenzen, ob dies
      tatschlich im berechtigten Interesse des Arbeitgebers liegt.
   - Wettbewerbs- und Nebenttigkeitsverbote oder Geheimhaltungspflichten
      so weit gefasst sind, dass ihre Grenze zur Unverhltnismigkeit nicht
       eindeutig erkennbar ist.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
```

```
{clause}

### Antwort ###
```

B.1.9. Vacation

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Der gesetzliche Mindesturlaub (i. d. R. 20 Werktage bei einer 5-Tage-
       Woche bzw. 24 Werktage bei einer 6-Tage-Woche) unterschritten wird.
   - Der Urlaubsverzicht (insbesondere auf den gesetzlichen Mindesturlaub
       oder dessen Abgeltung) im Voraus festgeschrieben wird oder die
       Vererblichkeit des Urlaubs-/Abgeltungsanspruchs ausgeschlossen wird.

   - Die Urlaubsbertragung und/oder Urlaubsabgeltung gesetzlich
       unzulssigen Beschrnkungen unterliegt (z. B. Verfall vor Ablauf der
       gesetzlich oder durch Rechtsprechung vorgegebenen Fristen, keine
       Bercksichtigung der 15-Monats-Frist bei Langzeiterkrankung).
   - Betriebsferien oder verpflichtend angeordneter Urlaub so geregelt
       werden, dass dem Arbeitnehmer kein eigener Gestaltungsspielraum fr
       seine restlichen Urlaubstage verbleibt und dadurch der gesetzliche
       Erholungszweck gefhrdet wird.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
   - Die Anzahl der Urlaubstage zwar oberhalb des Mindesturlaubs liegt,
       aber unklar bleibt, ob dies tatschlich den gesetzlichen
       Mindesturlaub sicherstellt (z. B. nicht eindeutig geklrt, ob es
       sich um Werk- oder Arbeitstage handelt).
   - Betriebsferien oder Zwangsurlaub in einem solchen Ausma vorgesehen
       sind, dass unklar ist, ob dem Arbeitnehmer genug eigener Spielraum
```

```
      bei der Urlaubsplanung verbleibt.
   - bertragungs- oder Antragsfristen (z. B. Anmeldung des Urlaubs bis zu
      einem sehr frhen Zeitpunkt, Fristen, die schneller verfallen als
      gesetzlich zulssig) vorgesehen sind, ohne hinreichende Flexibilitt
      oder hinreichende Beachtung gesetzlicher Regelungen.
   - Ein Rckgriff auf den gesetzlichen Urlaubsanspruch durch
      Zusatzvereinbarungen unklar oder intransparent geregelt wird (z. B.
       ob Zusatz- und Mindesturlaub unterschiedlich behandelt werden).

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.10. Compensation

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - berstunden pauschal mit dem Grundgehalt abgegolten werden, ohne eine
      angemessene Begrenzung.
```

- Vergtungsansprche des Arbeitnehmers nicht pfndbar oder abtretbar sind .
- Zuschlge fr Nacht-, Sonntags- oder Feiertagsarbeit pauschal ausgeschlossen oder mit dem Gehalt abgegolten werden.
- Die Vergtungshhe gegen den gesetzlichen Mindestlohn verstt.
- Schulungen oder Fortbildungen als unvergtete Arbeitszeit deklariert werden.
- Variable Vergtungsbestandteile ohne transparente und berprfbare Kriterien gendert oder gestrichen werden knnen.
- Auendienstmitarbeiter oder bestimmte Gruppen von Arbeitnehmern von der berstundenvergtung ausgeschlossen werden.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder unklar formuliert ist und mglicherweise gegen arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Die Vergtungshhe mglicherweise gegen den gesetzlichen Mindestlohn verstt.
   - Eine Rckzahlungspflicht fr Sonderzahlungen (z. B. Weihnachtsgeld) von der Dauer des Arbeitsverhltnisses abhngig gemacht wird.
   - Die vermgenswirksamen Leistungen (VWL) nicht frderfhig sind, weil die Hchstgrenze berschritten wird.
   - berstunden teilweise oder pauschal abgegolten werden, ohne dass klar geregelt ist, welche Arbeitszeiten betroffen sind.
   - Die Auszahlung von Mehrarbeitsvergtung von einer Genehmigung durch den Vorgesetzten abhngig gemacht wird.
   - Die Vergtung pauschal fr alle Ttigkeiten festgelegt wird, ohne Bercksichtigung von Mehrarbeit oder hher vergteten Ttigkeiten.
   - Eine nderung des Arbeitsgebiets oder der Funktion zu Einkommensverlusten fhren kann, ohne dass klare Schutzmechanismen existieren.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als 'unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere Erklrungen oder Text:

```
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.11. Statute of Limitations

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - berstunden pauschal mit dem Grundgehalt abgegolten werden, ohne eine
       angemessene Begrenzung.
   - Vergtungsansprche des Arbeitnehmers nicht pfndbar oder abtretbar sind
       .
   - Zuschlge fr Nacht-, Sonntags- oder Feiertagsarbeit pauschal
       ausgeschlossen oder mit dem Gehalt abgegolten werden.
   - Die Vergtungshhe gegen den gesetzlichen Mindestlohn verstt.
   - Schulungen oder Fortbildungen als unvergtete Arbeitszeit deklariert
       werden.
   - Variable Vergtungsbestandteile ohne transparente und berprfbare
       Kriterien gendert oder gestrichen werden knnen.
   - Auendienstmitarbeiter oder bestimmte Gruppen von Arbeitnehmern von
       der berstundenvergtung ausgeschlossen werden.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist der Fall, wenn:
   - Die Vergtungshhe mglicherweise gegen den gesetzlichen Mindestlohn
       verstt.
```

```
   - Eine Rckzahlungspflicht fr Sonderzahlungen (z. B. Weihnachtsgeld) von
       der Dauer des Arbeitsverhltnisses abhngig gemacht wird.
   - Die vermgenswirksamen Leistungen (VWL) nicht frderfhig sind, weil die
       Hchstgrenze berschritten wird.
   - berstunden teilweise oder pauschal abgegolten werden, ohne dass klar
       geregelt ist, welche Arbeitszeiten betroffen sind.
   - Die Auszahlung von Mehrarbeitsvergtung von einer Genehmigung durch
       den Vorgesetzten abhngig gemacht wird.
   - Die Vergtung pauschal fr alle Ttigkeiten festgelegt wird, ohne
       Bercksichtigung von Mehrarbeit oder hher vergteten Ttigkeiten.
   - Eine nderung des Arbeitsgebiets oder der Funktion zu
       Einkommensverlusten fhren kann, ohne dass klare Schutzmechanismen
       existieren.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.12. Penalty Clause

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.
```

```
### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - Die Vertragsstrafe unverhltnismig hoch ist und eine abschreckende
      Wirkung hat.
   - Ein Wettbewerbsverbot ber das gesetzlich zulssige Ma hinausgeht oder
      ohne Karenzentschdigung vereinbart wird.
   - Die Geheimhaltungspflicht zu weit gefasst ist und bekannte oder nicht
       vertrauliche Informationen umfasst.
   - Arbeitnehmer zur Verschwiegenheit ber ihr Gehalt verpflichtet werden.
   - Eine pauschale oder unbeschrnkte Nebenttigkeitsverbotsklausel besteht
      .
   - Rckforderungen von Schulungs- oder Einarbeitungskosten ohne
      abgestufte Regelung und Bercksichtigung der Betriebszugehrigkeit
      enthalten sind.
   - Vertragsstrafen fr allgemeine Verste ohne klare Einschrnkungen oder
      Prfung vorgesehen sind.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist der Fall, wenn:
   - Die Vertragsstrafe oder Sanktion unangemessen hoch ist oder nicht
      ausreichend begrenzt wird.
   - Ein Wettbewerbsverbot ohne angemessene Kompensationszahlung
      vereinbart wird.
   - Ein generelles Verbot von Nebenttigkeiten besteht, ohne eine
      Mitteilungspflicht oder Widerspruchsmglichkeit.
   - Eine Vertragsstrafe bei Nichterfllung der Kndigungsfrist vorgesehen
      ist, ohne vorherige Abmahnung.
   - Bei Dauerversten wiederholt Strafen anfallen, ohne eine klare
      Begrenzung.
   - Eine Rckforderung von Schulungs- oder Einarbeitungskosten vorgesehen
      ist, ohne abgestufte Rckzahlungsmodalitten.
   - Die Vertragsstrafe intransparent oder nicht nachvollziehbar
      formuliert ist.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.
```

```
4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

B.1.13. Overtime

```
### Aufgabe ###
Bewerte die folgende Klausel aus einem Arbeitsvertrag basierend auf den
    angegebenen Regeln als entweder 'unbedenklich', 'potentiell unzulssig'
     oder 'unzulssig'.

### Anweisungen ###
1. Eine Klausel ist 'unzulssig', wenn sie eindeutig gegen
    arbeitsrechtliche Vorgaben verstt. Das ist der Fall, wenn:
   - berstunden oder Mehrarbeit pauschal und unbegrenzt mit dem
       Grundgehalt abgegolten werden, ohne eine angemessene Begrenzung
       oder transparente Regelung.
   - Eine verpflichtende Leistung von berstunden angeordnet wird, ohne
       dass ein sachlicher Grund, tarifliche Ermchtigung oder eine
       Hchstgrenze vorgesehen ist (z. B. unbegrenzte berstundenpflicht
       ohne Rcksicht auf Arbeitszeitschutzvorschriften).
   - Ein vollstndiger Verzicht auf Vergtung oder Freizeitausgleich fr
       geleistete berstunden vereinbart wird, obwohl hier ein gesetzlicher
        Mindestschutz besteht.

2. Eine Klausel ist 'potentiell unzulssig', wenn sie problematisch oder
    unklar formuliert ist und mglicherweise gegen arbeitsrechtliche
    Vorgaben verstt. Das ist insbesondere der Fall, wenn:
```

```
   - Eine Pflicht zur Leistung von berstunden vorgesehen ist, aber nicht
     klar wird, unter welchen Voraussetzungen und in welchem Umfang
     diese angeordnet werden knnen (z. B. schwammige Begriffe wie nach
     Bedarf oder im Ermessen des Arbeitgebers ohne weitere Begrenzung).
   - Eine Vergtungs- oder Ausgleichsregelung fr berstunden zwar existiert,
      aber unzureichend oder missverstndlich geregelt ist (z. B. kein
     Hinweis auf gesetzliche oder tarifliche Hchstgrenzen, Unklarheit
     bezglich Vergtungshhe oder Zeitwertausgleich).
   - Die Klausel mgliche Rechte des Arbeitnehmers (z. B. Ablehnung von
     berstunden aus familiren oder gesundheitlichen Grnden) nicht
     bercksichtigt oder sehr vage einschrnkt.

3. Falls keine dieser Regeln verletzt wird, muss die Klausel als '
    unbedenklich' bewertet werden.

4. Falls du dir nicht sicher bist oder keine klare Regelverletzung
    vorliegt, entscheide dich fr 'unbedenklich'.

5. Antworte nur mit einer der drei folgenden Optionen, ohne weitere
    Erklrungen oder Text:
   - 'unbedenklich'
   - 'potentiell unzulssig'
   - 'unzulssig'

### Klausel ###
{clause}

### Antwort ###
```

List of Figures

List of Tables

Bibliography

[1] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, *A survey of large language models*, 2024. arXiv: 2303.18223 [cs.CL].

[2] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao, *Large language models: A survey*, 2024. arXiv: 2402.06196 [cs.CL].

[3] Z. Sun, "A short survey of viewing large language models in legal aspect," *arXiv preprint arXiv:2303.09136*, 2023.

[4] B. Padiu, R. Iacob, T. Rebedea, and M. Dascalu, "To what extent have llms reshaped the legal domain so far? a scoping literature review," *Information*, vol. 15, no. 11, p. 662, 2024.

[5] H. Li, W. Su, C. Wang, Y. Wu, Q. Ai, and Y. Liu, "Thuir@ coliee 2023: Incorporating structural knowledge into pre-trained language models for legal case retrieval," *arXiv preprint arXiv:2305.06812*, 2023.

[6] A. Louis, G. van Dijck, and G. Spanakis, "Interpretable long-form legal question answering with retrieval-augmented large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, 2024, pp. 22 266–22 275.

[7] J. Guan, Z. Yu, Y. Liao, R. Tang, M. Duan, and G. Han, "Predicting critical path of labor dispute resolution in legal domain by machine learning models based on shapley additive explanations and soft voting strategy," *Mathematics*, vol. 12, no. 2, p. 272, 2024.

[8] D. H. Sleeman and K. Gilhooly, *Groups of experts often differ in their decisions: What are the implications for ai and machine learning? a commentary on noise: A flaw in human judgment, by kahneman, sibony, and sunstein (2021)*, 2023.

[9] O. Weller, B. Chang, S. MacAvaney, K. Lo, A. Cohan, B. Van Durme, D. Lawrie, and L. Soldaini, "Followir: Evaluating and teaching information retrieval models to follow instructions," *arXiv preprint arXiv:2403.15246*, 2024.

[10] J. Savelka and K. D. Ashley, "The unreasonable effectiveness of large language models in zero-shot semantic annotation of legal texts," *Frontiers in Artificial Intelligence*, vol. 6, p. 1 279 794, 2023.

[11] S. Meisenbacher, *Ai-assisted legal analysis and correction of german employment contracts*, Website of the Technical University of Munich, Last modified: July 23, 2024, 2024.

[12] J. Lai, W. Gan, J. Wu, Z. Qi, and S. Y. Philip, "Large language models in law: A survey," *AI Open*, 2024.

[13] S. Villata, M. Araszkiewicz, K. Ashley, T. Bench-Capon, L. K. Branting, J. G. Conrad, and A. Wyner, "Thirty years of artificial intelligence and law: The third decade," *Artificial Intelligence and Law*, vol. 30, no. 4, pp. 561–591, 2022.

[14] Z. Fei, X. Shen, D. Zhu, F. Zhou, Z. Han, S. Zhang, K. Chen, Z. Shen, and J. Ge, "Lawbench: Benchmarking legal knowledge of large language models," *arXiv preprint arXiv:2309.16289*, 2023.

[15] V. Magesh, F. Surani, M. Dahl, M. Suzgun, C. D. Manning, and D. E. Ho, "Hallucination-free? assessing the reliability of leading ai legal research tools," *arXiv preprint arXiv:2405.20362*, 2024.

[16] Q. Huang, M. Tao, C. Zhang, Z. An, C. Jiang, Z. Chen, Z. Wu, and Y. Feng, "Lawyer llama technical report," *arXiv preprint arXiv:2305.15062*, 2023.

[17] J. Cui, Z. Li, Y. Yan, B. Chen, and L. Yuan, "Chatlaw: Open-source legal large language model with integrated external knowledge bases," *CoRR*, 2023.

[18] N. Wiratunga, R. Abeyratne, L. Jayawardena, K. Martin, S. Massie, I. Nkisi-Orji, R. Weerasinghe, A. Liret, and B. Fleisch, "Cbr-rag: Case-based reasoning for retrieval augmented generation in llms for legal question answering," in *International Conference on Case-Based Reasoning*, Springer, 2024, pp. 445–460.

[19] N. Pipitone and G. H. Alami, "Legalbench-rag: A benchmark for retrieval-augmented generation in the legal domain," *arXiv preprint arXiv:2408.10343*, 2024.

[20] X. Peng and L. Chen, "Athena: Retrieval-augmented legal judgment prediction with large language models," *arXiv preprint arXiv:2410.11195*, 2024.

[21] D. Trautmann, A. Petrova, and F. Schilder, "Legal prompt engineering for multilingual legal judgement prediction," *arXiv preprint arXiv:2212.02199*, 2022.

[22] F. Yu, L. Quartey, and F. Schilder, "Legal prompting: Teaching a language model to think like a lawyer," *arXiv preprint arXiv:2212.01326*, 2022.

[23] S. Schulhoff, M. Ilie, N. Balepur, K. Kahadze, A. Liu, C. Si, Y. Li, A. Gupta, H. Han, S. Schulhoff, *et al.*, "The prompt report: A systematic survey of prompting techniques," *arXiv preprint arXiv:2406.06608*, 2024.

[24] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large language models are human-level prompt engineers," *arXiv preprint arXiv:2211.01910*, 2022.

[25] X. Wan, R. Sun, H. Nakhost, and S. O. Arik, "Teach better or show smarter? on instructions and exemplars in automatic prompt optimization," *arXiv preprint arXiv:2406.15708*, 2024.

[26] T. Sun, Z. He, H. Qian, Y. Zhou, X. Huang, and X. Qiu, "Bbtv2: Towards a gradient-free future with large language models," *arXiv preprint arXiv:2205.11200*, 2022.

[27] T. Sun, Y. Shao, H. Qian, X. Huang, and X. Qiu, "Black-box tuning for language-model-as-a-service," in *International Conference on Machine Learning*, PMLR, 2022, pp. 20 841–20 855.

[28] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, *et al.*, "Lora: Low-rank adaptation of large language models.," *ICLR*, vol. 1, no. 2, p. 3, 2022.

[29] Z. Zhong, D. Friedman, and D. Chen, "Factual probing is [mask]: Learning vs. learning to recall," *arXiv preprint arXiv:2104.05240*, 2021.

[30] Y. Wen, N. Jain, J. Kirchenbauer, M. Goldblum, J. Geiping, and T. Goldstein, "Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[31] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[32] M. Deng, J. Wang, C.-P. Hsieh, Y. Wang, H. Guo, T. Shu, M. Song, E. P. Xing, and Z. Hu, "Rlprompt: Optimizing discrete text prompts with reinforcement learning," *arXiv preprint arXiv:2205.12548*, 2022.

[33] L. Chen, J. Chen, T. Goldstein, H. Huang, and T. Zhou, "Instructzero: Efficient instruction optimization for black-box large language models," *arXiv preprint arXiv:2306.03082*, 2023.

[34] J. De Curtò, I. de Zarzà, G. Roig, J. C. Cano, P. Manzoni, and C. T. Calafate, "Llm-informed multi-armed bandit strategies for non-stationary environments," *Electronics*, vol. 12, no. 13, p. 2814, 2023.

[35] T. Liu, N. Astorga, N. Seedat, and M. van der Schaar, "Large language models to enhance bayesian optimization," *arXiv preprint arXiv:2402.03921*, 2024.

[36] Q. Guo, R. Wang, J. Guo, B. Li, K. Song, X. Tan, G. Liu, J. Bian, and Y. Yang, "Connecting large language models with evolutionary algorithms yields powerful prompt optimizers," *arXiv preprint arXiv:2309.08532*, 2023.

[37] C. Fernando, D. Banarse, H. Michalewski, S. Osindero, and T. Rocktäschel, "Promptbreeder: Self-referential self-improvement via prompt evolution," *arXiv preprint arXiv:2309.16797*, 2023.

[38] R. Pryzant, D. Iter, J. Li, Y. T. Lee, C. Zhu, and M. Zeng, "Automatic prompt optimization with" gradient descent" and beam search," *arXiv preprint arXiv:2305.03495*, 2023.

[39] X. Wang, C. Li, Z. Wang, F. Bai, H. Luo, J. Zhang, N. Jojic, E. P. Xing, and Z. Hu, "Promptagent: Strategic planning with language models enables expert-level prompt optimization," *arXiv preprint arXiv:2310.16427*, 2023.

[40] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen, *Large language models as optimizers*, 2024. arXiv: 2309.03409 [cs.LG].

[41] O. Wardas and F. Matthes, "Ai-assisted german employment contract review: A benchmark dataset," *arXiv preprint arXiv:2501.17194*, 2025.

[42] Y. Sun, J.-B. Tien, *et al.*, "Retrieval augmented prompt optimization," in *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

[43] W. Qin and Z. Sun, "Exploring the nexus of large language models and legal systems: A short survey," *arXiv preprint arXiv:2404.00990*, 2024.

[44] A. Blair-Stanek, N. Holzenberger, and B. Van Durme, "Can gpt-3 perform statutory reasoning?" In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, 2023, pp. 22–31.

[45] C. Nguyen and L.-M. Nguyen, "Employing label models on chatgpt answers improves legal text entailment performance," *arXiv preprint arXiv:2401.17897*, 2024.

[46] T. Nguyen and E. Wong, "In-context example selection with influences," *arXiv preprint arXiv:2302.11042*, 2023.

[47] Y. Wu, S. Zhou, Y. Liu, W. Lu, X. Liu, Y. Zhang, C. Sun, F. Wu, and K. Kuang, "Precedent-enhanced legal judgment prediction with llm and domain-model collaboration," *arXiv preprint arXiv:2310.09241*, 2023.

[48] A. Deroy, K. Ghosh, and S. Ghosh, "How ready are pre-trained abstractive models and llms for legal case judgement summarization?" *arXiv preprint arXiv:2306.01248*, 2023.

[49] J. Savelka, K. D. Ashley, M. A. Gray, H. Westermann, and H. Xu, "Explaining legal concepts with augmented large language models (gpt-4)," *arXiv preprint arXiv:2306.09525*, 2023.

[50] Y. Zhou, H. Huang, and Z. Wu, "Boosting legal case retrieval by query content selection with large language models," in *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, 2023, pp. 176–184.

[51] R. Macey-Dare, "Chatgpt & generative ai systems as quasi-expert legal advice lawyers-case study considering potential appeal against conviction of tom hayes," *Available at SSRN 4342686*, 2023.

[52] K. Y. Iu and V. M.-Y. Wong, "Chatgpt by openai: The end of litigation lawyers?" *Available at SSRN 4339839*, 2023.

[53] J. Lee, J.-S. Yi, and J. Son, "Development of automatic-extraction model of poisonous clauses in international construction contracts using rule-based nlp," *Journal of Computing in Civil Engineering*, vol. 33, no. 3, p. 04 019 003, 2019.

[54] D. Hendrycks, C. Burns, A. Chen, and S. Ball, "Cuad: An expert-annotated nlp dataset for legal contract review," *arXiv preprint arXiv:2103.06268*, 2021.

[55] S. Leivaditi, J. Rossi, and E. Kanoulas, "A benchmark for lease contract review," *arXiv preprint arXiv:2010.10386*, 2020.

[56] M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, *et al.*, "Challenging big-bench tasks and whether chain-of-thought can solve them," *arXiv preprint arXiv:2210.09261*, 2022.

[57] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Advances in neural information processing systems*, vol. 13, 2000.

[58] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[59] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, Makuhari, vol. 2, 2010, pp. 1045–1048.

[60] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, *et al.*, "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt," *International Journal of Machine Learning and Cybernetics*, pp. 1–65, 2024.

[61] X. Han, Z. Zhang, N. Ding, Y. Gu, X. Liu, Y. Huo, J. Qiu, Y. Yao, A. Zhang, L. Zhang, *et al.*, "Pre-trained models: Past, present and future," *AI Open*, vol. 2, pp. 225–250, 2021.

[62] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," *Science China technological sciences*, vol. 63, no. 10, pp. 1872–1897, 2020.

[63] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[64] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[65] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[66] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark, *et al.*, "Training compute-optimal large language models," *arXiv preprint arXiv:2203.15556*, 2022.

[67] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[68] A. Rao, J. Kim, M. Kamineni, M. Pang, W. Lie, and M. D. Succi, "Evaluating chatgpt as an adjunct for radiologic decision-making," *MedRxiv*, pp. 2023–02, 2023.

[69] S. Montagna, S. Ferretti, L. C. Klopfenstein, A. Florio, and M. F. Pengo, "Data decentralisation of llm-based chatbot systems in chronic disease self-management," in *Proceedings of the 2023 ACM Conference on Information Technology for Social Good*, 2023, pp. 205–212.

[70] G. Deiana, M. Dettori, A. Arghittu, A. Azara, G. Gabutti, and P. Castiglia, "Artificial intelligence and public health: Evaluating chatgpt responses to vaccination myths and misconceptions," *Vaccines*, vol. 11, no. 7, p. 1217, 2023.

[71] W. Dai, J. Lin, H. Jin, T. Li, Y.-S. Tsai, D. Gašević, and G. Chen, "Can large language models provide feedback to students? a case study on chatgpt," in *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, IEEE, 2023, pp. 323–325.

[72] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier, *et al.*, "Chatgpt for good? on opportunities and challenges of large language models for education," *Learning and individual differences*, vol. 103, p. 102 274, 2023.

[73] J. C. Young and M. Shishido, "Investigating openai's chatgpt potentials in generating chatbot's dialogue for english as a foreign language learning," *International journal of advanced computer science and applications*, vol. 14, no. 6, 2023.

[74] J. Drápal, H. Westermann, J. Savelka, *et al.*, "Using large language models to support thematic analysis in empirical legal studies.," in *JURIX*, 2023, pp. 197–206.

[75] N. Guha, J. Nyarko, D. Ho, C. Ré, A. Chilton, A. Chohlas-Wood, A. Peters, B. Waldon, D. Rockmore, D. Zambrano, *et al.*, "Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[76] N. Maus, P. Chao, E. Wong, and J. Gardner, "Black box adversarial prompting for foundation models," *arXiv preprint arXiv:2302.04237*, 2023.

[77] H. Yang, X.-Y. Liu, and C. D. Wang, "Fingpt: Open-source financial large language models," *arXiv preprint arXiv:2306.06031*, 2023.

[78] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 2021, pp. 610–623.

[79] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, *et al.*, "Siren's song in the ai ocean: A survey on hallucination in large language models," *arXiv preprint arXiv:2309.01219*, 2023.

[80] B. Meskó, "Prompt engineering as an important emerging skill for medical professionals: Tutorial," *Journal of medical Internet research*, vol. 25, e50638, 2023.

[81] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *arXiv preprint arXiv:2302.11382*, 2023.

[82] T. F. Heston and C. Khun, "Prompt engineering in medical education," *International Medical Education*, vol. 2, no. 3, pp. 198–205, 2023.

[83] M. U. Hadi, Q. Al Tashi, A. Shah, R. Qureshi, A. Muneer, M. Irfan, A. Zafar, M. B. Shaikh, N. Akhtar, J. Wu, *et al.*, "Large language models: A comprehensive survey of its applications, challenges, limitations, and future prospects," *Authorea Preprints*, 2024.

[84] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, Y. Zhang, N. Zhenqiang Gong, *et al.*, "Promptbench: Towards evaluating the robustness of large language models on adversarial prompts," *arXiv e-prints*, arXiv–2306, 2023.

[85] J. Zamfirescu-Pereira, R. Y. Wong, B. Hartmann, and Q. Yang, "Why johnny can't prompt: How non-ai experts try (and fail) to design llm prompts," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–21.

[86] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[87] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.

[88] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[89] J. Long, "Large language model guided tree-of-thought," *arXiv preprint arXiv:2305.08291*, 2023.

[90] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[91] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, "A systematic survey of prompt engineering in large language models: Techniques and applications," *arXiv preprint arXiv:2402.07927*, 2024.

[92] R. Wang, S. An, M. Cheng, T. Zhou, S. J. Hwang, and C.-J. Hsieh, "Mixture-of-experts in prompt optimization,"

[93] X. L. Do, Y. Zhao, H. Brown, Y. Xie, J. X. Zhao, N. F. Chen, K. Kawaguchi, M. Shieh, and J. He, "Prompt optimization via adversarial in-context learning," *arXiv preprint arXiv:2312.02614*, 2023.

[94] T. Zhang, X. Wang, D. Zhou, D. Schuurmans, and J. E. Gonzalez, "Tempera: Test-time prompting via reinforcement learning," *arXiv preprint arXiv:2211.11890*, 2022.

[95] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[96] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?" *arXiv preprint arXiv:2101.06804*, 2021.

[97] O. Rubin, J. Herzig, and J. Berant, "Learning to retrieve prompts for in-context learning," *arXiv preprint arXiv:2112.08633*, 2021.

[98] B. Xu, Q. Wang, Z. Mao, Y. Lyu, Q. She, and Y. Zhang, "k nn prompting: Beyond-context learning with calibration-free nearest neighbor inference," *arXiv preprint arXiv:2303.13824*, 2023.

[99] Y. Lu, M. Bartolo, A. Moore, S. Riedel, and P. Stenetorp, "Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity," *arXiv preprint arXiv:2104.08786*, 2021.

[100] Z. Wu, Y. Wang, J. Ye, and L. Kong, "Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering," *arXiv preprint arXiv:2212.10375*, 2022.

[101] Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, "Calibrate before use: Improving few-shot performance of language models," in *International conference on machine learning*, PMLR, 2021, pp. 12 697–12 706.

[102] Y. Chen, C. Zhao, Z. Yu, K. McKeown, and H. He, "On the relation between sensitivity and accuracy in in-context learning," *arXiv preprint arXiv:2209.07661*, 2022.

[103] J. Ye, Z. Wu, J. Feng, T. Yu, and L. Kong, "Compositional exemplars for in-context learning," in *International Conference on Machine Learning*, PMLR, 2023, pp. 39 818–39 833.

[104] Y. Zhang, S. Feng, and C. Tan, "Active example selection for in-context learning," *arXiv preprint arXiv:2211.04486*, 2022.

[105] X. Wan, R. Sun, H. Dai, S. O. Arik, and T. Pfister, "Better zero-shot reasoning with self-adaptive prompting," *arXiv preprint arXiv:2305.14106*, 2023.

[106] X. Wan, R. Sun, H. Nakhost, H. Dai, J. M. Eisenschlos, S. O. Arik, and T. Pfister, "Universal self-adaptive prompting," *arXiv preprint arXiv:2305.14926*, 2023.

[107] H. J. Kim, H. Cho, J. Kim, T. Kim, K. M. Yoo, and S.-g. Lee, "Self-generated in-context learning: Leveraging auto-regressive language models as a demonstration generator," *arXiv preprint arXiv:2206.08082*, 2022.

[108] K. Margatina, T. Schick, N. Aletras, and J. Dwivedi-Yu, "Active learning principles for in-context learning with large language models," *arXiv preprint arXiv:2305.14264*, 2023.

[109] O. Honovich, U. Shaham, S. R. Bowman, and O. Levy, "Instruction induction: From few examples to natural language task descriptions," *arXiv preprint arXiv:2205.10782*, 2022.

[110] W. Cui, J. Zhang, Z. Li, H. Sun, D. Lopez, K. Das, B. Malin, and S. Kumar, "Phaseevo: Towards unified in-context prompt optimization for large language models," *arXiv preprint arXiv:2402.11347*, 2024.

[111] M. Li, W. Wang, F. Feng, Y. Cao, J. Zhang, and T.-S. Chua, "Robust prompt optimization for large language models against distribution shifts," *arXiv preprint arXiv:2305.13954*, 2023.

[112] Y. Guo, Y. Liang, C. Wu, W. Wu, D. Zhao, and N. Duan, "Learning to plan with natural language," *arXiv preprint arXiv:2304.10464*, 2023.

[113] Q. Ye, M. Axmed, R. Pryzant, and F. Khani, "Prompt engineering a prompt engineer," *arXiv preprint arXiv:2311.05661*, 2023.

[114] G. Juneja, N. Natarajan, H. Li, J. Jiao, and A. Sharma, "Task facet learning: A structured approach to prompt optimization," *arXiv preprint arXiv:2406.10504*, 2024.

[115] S. Srivastava, C. Huang, W. Fan, and Z. Yao, "Instances need more care: Rewriting prompts for instances with llms in the loop yields better zero-shot performance," in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 6211–6232.

[116] Y. Dong, K. Luo, X. Jiang, Z. Jin, and G. Li, "Pace: Improving prompt with actor-critic editing for large language model," *arXiv preprint arXiv:2308.10088*, 2023.

[117] M. Diesendruck, J. Lin, S. Imani, G. Mahalingam, M. Xu, and J. Zhao, "Learning how to ask: Cycle-consistency refines prompts in multimodal foundation models," *arXiv preprint arXiv:2402.08756*, 2024.

[118] J. G. Billa, M. Oh, and L. Du, "Supervisory prompt training," *arXiv preprint arXiv:2403.18051*, 2024.

[119] A. Prasad, P. Hase, X. Zhou, and M. Bansal, "Grips: Gradient-free, edit-based instruction search for prompting large language models," *arXiv preprint arXiv:2203.07281*, 2022.

[120] C.-J. Hsieh, S. Si, F. X. Yu, and I. S. Dhillon, "Automatic engineering of long prompts," *arXiv preprint arXiv:2311.10117*, 2023.

[121] X. Tang, X. Wang, W. Zhao, S. Lu, Y. Li, and J. Wen, *Unleashing the potential of large language models as prompt optimizers: An analogical analysis with gradient-based model optimizers. corr abs/2402.17564(2024)*.

[122] R. Ma, X. Wang, X. Zhou, J. Li, N. Du, T. Gui, Q. Zhang, and X. Huang, "Are large language models good prompt optimizers?" *arXiv preprint arXiv:2402.02101*, 2024.

[123] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019, pp. 3982–3992.

[124] OpenAI, *Text-embedding-ada-002 model documentation*, `https://platform.openai.com/docs/guides/embeddings`, 2022.

[125] OpenAI, *Gpt-4o mini: Advancing cost-efficient intelligence*, Accessed: 12.12.2024, 2024.

[126] OpenAI, *Gpt-4o system card*, Accessed: 21.01.2025, 2024.

[127] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[128] L. Reynolds and K. McDonell, "Prompt programming for large language models: Beyond the few-shot paradigm," in *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, 2021, pp. 1–7.

[129] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," *Advances in neural information processing systems*, vol. 28, 2015.

[130] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (mlops): Overview, definition, and architecture," *IEEE access*, vol. 11, pp. 31 866–31 879, 2023.

[131] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.

[132] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[133] B. Settles, "Active learning literature survey," 2009.

[134] H. Zhou, X. Wan, Y. Liu, N. Collier, I. Vulić, and A. Korhonen, "Fairer preferences elicit improved human-aligned large language model judgments," *arXiv preprint arXiv:2406.11370*, 2024.

[135] K. M. Richmond, S. M. Muddamsetty, T. Gammeltoft-Hansen, H. P. Olsen, and T. B. Moeslund, "Explainable ai and law: An evidential survey," *Digital Society*, vol. 3, no. 1, p. 1, 2024.

[136] M. C. Buiten, "Towards intelligent regulation of artificial intelligence," *European Journal of Risk Regulation*, vol. 10, no. 1, pp. 41–59, 2019.

[137] B.-b. authors, "Beyond the imitation game: Quantifying and extrapolating the capabilities of language models," *Transactions on Machine Learning Research*, 2023.

[138] X. Ying, "An overview of overfitting and its solutions," in *Journal of physics: Conference series*, IOP Publishing, vol. 1168, 2019, p. 022 022.

[139] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[140] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation," *Expert Systems with Applications*, vol. 244, p. 122 778, 2024.

[141] R. Artstein and M. Poesio, "Inter-coder agreement for computational linguistics," *Computational linguistics*, vol. 34, no. 4, pp. 555–596, 2008.

[142] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 8, no. 4, e1249, 2018.

[143] S. Chen, B. Li, and D. Niu, "Boosting of thoughts: Trial-and-error problem solving with large language models," *arXiv preprint arXiv:2402.11140*, 2024.

[144] B. Hou, J. O'connor, J. Andreas, S. Chang, and Y. Zhang, "Promptboosting: Black-box text classification with ten forward passes," in *International Conference on Machine Learning*, PMLR, 2023, pp. 13 309–13 324.

[145] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, pp. 123–140, 1996.

[146] R. E. Schapire, "The strength of weak learnability," *Machine learning*, vol. 5, pp. 197–227, 1990.

[147] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[148] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International conference on machine learning*, PMLR, 2019, pp. 2790–2799.

[149] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[150] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang, *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024.