universität
uulm

Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Datenbanken und Informationssysteme (DBIS)

Identifikation von DSGVO-kritischen Aktivitäten in Business Prozessen mittels Large Language Models

Abschlussarbeit an der Universität Ulm

Vorgelegt von:
Merten Dieckmann
merten.dieckmann@uni-ulm.de
1058340

Gutachter:
Prof. Dr. Manfred Reichert
Prof. Dr. Rüdiger Pryss

Betreuer:
Magdalena von Schwerin

2025

Fassung 22. September 2025

Satz: PDF-LaTeX 2ε

Inhaltsverzeichnis

Abkürzungen

DSGVO Datenschutz-Grundverordnung
LM Language Model
LLM Large Language Model
RAG Retrieval Augmented Generation
BPMN Business Process Model and Notation
EU Europäische Union

1 Einleitung

1.1 Motivation

- Geschäftsprozesse fast in allen Organisationen allgegenwärtig und bilden die Grundlage für effiziente Abläufe
- Außerdem ist durch die Datenschutz-Grundverordnung (DSGVO) in Europa Datenschutz ein zentraler regulatorischer Aspekt [7, 9]. Unternehmen müssen sicherstellen, dass in Prozessen personenbezogene Daten rechtskonform verarbeitet werden. Ansonsten drohen Strafen von bis zu 20 Millionen Euro oder 4% des weltweiten gesamten erzielten Jahresumsatzes [9].
- Die Überprüfung von Prozessen auf konformität in Bezug auf Datenschutz ist jedoch zeit- und kostenintensiv [17, 28]. Besonders in großen Organisationen mit hunderten von Prozessen parallel ist eine manuelle Analyse nicht wirklich praktikabel und fehleranfällig. Fehlerhafte Unterekennung datenschutzkritischer Aktivitäten, also False Negatives, kann weitreichende Folgen haben - von Reputationsschäden bis hin zu hohen Bußgeldern [17].
- Large Language Models (LLMs) versprechen genau an dieser Stelle eine Lösung, indem sie ein schnelles automatisiertes Screening von Prozessmodellen ermöglichen. Erste Arbeiten zeigen das Potenzial von LLMs, etwa bei der Identifikation datenschutzrelevanter Verarbeitungstätigkeiten oder in der Analyse von Datenschutzerklärungen [8, 23]. Besonders interessant sind dabei europäische Open-Source Modelle wie die von Mistral [2]. Sie sind zum einen frei verfügbar und transparent und zum anderen sind sie bislang noch kaum in Hinblick auf DSGVO-bezogene Aufgaben evaluiert worden. Bisher fehlen belastbare empirische Vergleiche und reproduzierbare Evidenz, die eine fundierte Bewertung dieser Modelle erlauben würden [27].

1.2 Problemstellung

- Trotz der genannten Potenziale fehlt es bisher an standardisierten, reproduzierbaren Vergleichen verschiedener Modelle für die konkrete Aufgabe Aktivitäten in Geschäftsprozessen nach „kritisch“ und „unkritisch“ zu klassifizieren. Erste Ansätze, wie z.B. der von Nake et al. [17], zeigen dass maschinelles Lernen grundsätzlich in der Lage ist DSGVO-kritische Aktivitäten in textuellen Prozessbeschreibungen zu erkennen, jedoch existieren keine einheitlichen Benchmarks für einen systematischen Vergleich unterschiedlicher LLMs.
- Auch von Schwerin und Reichert [27] heben hervor, dass trotz großer Fortschritte im Einsatz von LLMs für juristische Aufgaben bislang erhebliche Lücken in der Evaluation für compliance-spezifische Anwendungen bestehen und geeignete DSGVO-spezifische Benchmarks fehlen.
- Besonders interessant ist die Frage, wie sich Open-Source-Modelle - insbesondere mit Ursprung aus der Europäische Union (EU) - im Vergleich zu internationalen außerhalb der EU entwickelten Modellen schlagen und welche Trade-offs dabei entstehen [27].
- TODO Eine zusätzliche Herausforderung ergibt sich aus der Natur von Business Process Model and Notation (BPMN)-Modellen: Typischerweise konzentrieren sie sich auf den Kontrollfluss und vernachlässigen die Datenebene. Datenobjekte werden oftmals gar nicht explizit modelliert oder nur implizit in den Aktivitäten referenziert. Dadurch ist die Datennutzung von Aktivitäten nicht direkt erkennbar und muss aus textuellen Beschreibungen und dem Kontext erschlossen werden [25]. Das erschwert dies die automatische Identifikation von DSGVO-kritischen Aktivitäten, da Algorithmen personenbezogene Datenflüsse zunächst indirekt und über den Kontext ableiten müssen.

1.3 Zielsetzung und Beiträge

Ziel der Arbeit ist es, einen methodischen Beitrag zur automatisierten Identifikation von DSGVO-kritischen Aktivitäten in Geschäftsprozessen zu leisten. Hierfür werden folgende Beiträge angestrebt:

- Entwicklung einer Klassifizierungspipeline für Geschäftsprozesse, die Aktivitäten binär in datenschutzkritisch oder unkritisch einordnet.
- Konzeption und Umsetzung eines Evaluationsframeworks, das reproduzierbare Vergleiche verschiedener LLMs und Algorithmen über eine einheitliche Schnittstelle ermöglicht.
- Entwicklung einer Labelingsoftware zur Erstellung und Annotation von Datensätzen für das Evaluationsframework.
- Aufbau eines repräsentativen Datensatzes aus gelabelten BPMN-Prozessen, inklusive klar definierter Labeling-Kriterien.
- Bereitstellung überprüfbarer empirischer Befunde, inklusive Code, Konfigurationen der Experimente und Seeds, um Nachvollziehbarkeit und Reproduzierbarkeit zu gewährleisten.

1.4 Forschungsfrage und Unterfragen

Die zentrale Forschungsfrage dieser Arbeit lautet:

FF1: Wie zuverlässig identifizieren LLMs DSGVO-kritische Aktivitäten in BPMN-Prozessmodellen?

Um diese Frage differenziert beantworten zu können werden außerdem folgende Unterfragen betrachtet:

- *UF1: Wie gut schneiden europäische Open-Source-Modelle im Vergleich zu internationalen Modellen ab?*
- *UF2: Wie unterscheiden sich große und kleine Modelle in ihrer Leistungsfähigkeit?*
- *UF3: Welche Open-Source-Modelle (insbesondere aus der EU) erzielen die besten Ergebnisse?*
- *UF4: Wie gut schneiden Open-Source-Modelle gegenüber kommerziellen Modellen wie GPT-4o ab?*

Für ein initiales Screening reicht, wie in [17], eine binäre Klassifikation (kritisch vs. unkritisch). Eine tiefergehende rechtliche Prüfung kann in einem nachfolgendem Schritt durchgeführt werden und nicht Bestandteil dieser Arbeit.

1.5 Aufbau der Arbeit

Die Arbeit ist wie folgt gegliedert: Kapitel 2 gibt einen Überblick über den theoretischen Hintergrund, die DSGVO und BPMN sowie eine Einführung in LLMs und verwandte Arbeiten. Kapitel 3 beschreibt den Rahmen der Entwicklung der Klassifizierungspipeline, des Evaluationsframeworks und der Experimente. Kapitel 4 stellt den entwickelten Algorithmus zur Klassifikation von BPMN-Modellen und dessen einheitliche Schnittstelle vor. Kapitel 5 präsentiert die Architektur und den Funktionsumfang der Evaluationspipeline. Anschließend wird in Kapitel 6 die Labelingsoftware und die Erstellung der Datensätze erläutert. Kapitel 7 zeigt auf wie die Auwahl der LLMs erfolgte. Kapitel 8 erläutert den Versuchsaufbau, Kapitel 9 die Durchführung der Experimente und Kapitel 10 stellt die Ergebnisse vor. In Kapitel 11 werden die Ergebnisse im Kontext der Forschungsfragen diskutiert. Zum Schluss fasst Kapitel 12 die Arbeit zusammen und Kapitel 13 gibt einen Ausblick auf mögliche zukünftige Forschungsthemen.

2 Hintergrund und verwandte Arbeiten

2.1 Datenschutzgrundverordnung (DSGVO)

Die europäische Datenschutz-Grundverordnung (DSGVO) [9] bildet den zentralen rechtlichen Rahmen für den Schutz personenbezogener Daten in der EU. Sie gilt seit dem 25. Mai 2018. Durch die DSGVO werden Betroffenenrechte gestärkt und Verantwortliche zu technischen und organisatorischen Maßnahmen verpflichtet, wie z. B. *Datenschutz durch Technikgestaltung* und *datenschutzfreundliche Voreinstellungen* (Art. 25 DSGVO) [10].

Definitionen

Im Folgenden werden zentrale Begriffe der DSGVO erläutert, die für das Verständnis dieser Arbeit relevant sind:

- **Personenbezogene Daten** sind alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person beziehen (Art. 4 Abs. 1 DSGVO) [9]. Eine Person ist identifizierbar, wenn sie direkt oder indirekt bestimmbar ist (z. B. anhand von Name, Kennnummer, Standortdaten, Online-Kennung).
- **Verarbeitung** bezeichnet *jeden* mit personenbezogenen Daten vorgenommenen Vorgang (Art. 4 Abs. 2 DSGVO) und umfasst insbesondere das **Erheben**, **Speichern**, **Verwenden**/**Nutzen**, **Offenlegen durch Übermittlung** sowie das **Löschen**/**Vernichten** [9].
- Im BPMN-Kontext sind alle Aktivitäten als **datenschutzkritisch** zu betrachten, die solche Verarbeitungshandlungen an personenbezogenen Daten vor-

nehmen oder auslösen (z. B. Abruf aus einem Kundendaten-Speicher, Übergabe an externe Stellen).

Abgrenzung: Risiko-Screening vs. Rechtsberatung

Die in dieser Arbeit eingesetzten Klassifizierungsverfahren dienen einem *automatisierten Risiko-Vorscreening* von Prozessaktivitäten. Sie ersetzen keine individuelle Rechtsprüfung im Einzelfall. Insbesondere in Deutschland ist die Erbringung konkreter Rechtsdienstleistungen Personen mit entsprechender Befugnis vorbehalten [6]. Die Ergebnisse sind daher als Entscheidungshilfe zu verstehen und bedürfen - insbesondere bei Grenzfällen - der Bewertung durch qualifizierte Experten.

2.2 Business Process Model and Notation (BPMN)

BPMN ist ein Standard zur Modellierung von Geschäftsprozessen. Die Notation wurde entwickelt, um eine einheitliche Notation bereitzustellen, die sowohl von Geschäftsanalysten als auch von technischen Entwicklern verstanden wird. BPMN-Modelle bestehen aus verschiedenen Elementen wie Aktivitäten, Ereignissen, Gateways und Verbindungen, die zusammen den Ablauf eines Geschäftsprozesses darstellen [18].

Relevante BPMN-Elemente

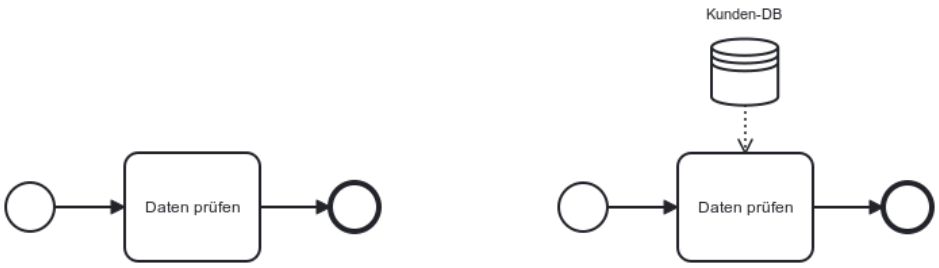
Für die Identifikation von DSGVO-kritischen Aktivitäten sind insbesondere folgende Elemente relevant, da sie Hinweise auf den Umgang mit (personenbezogenen) Daten geben:

- **Aktivitäten**: bilden die auszuführenden Arbeitsschritte eines Prozesses ab. Sie können Ein- und Ausgaben sowie Datenabhängigkeiten definieren [18]. Durch ihren Namen oder Kontext können Rückschlüsse auf die Verarbeitung personenbezogener Daten gezogen werden.

- **Sequenzflüsse**: verbinden Aktivitäten, Ereignisse und Gateways und zeigen die Reihenfolge der Ausführung im Prozess an [18]. Mit ihrer Hilfe kann eine einzelne Aktivität im Kontext des gesamten Prozesses betrachtet werden, indem der Pfad zu und von der Aktivität verfolgt wird.
- **Datenobjekte und Datenspeicher**: repräsentieren flüchtige oder persistente Daten, die im Prozess von z.B. Aktivitäten genutzt oder geschrieben werden können [18]. Sie können auch personenbezogene Daten enthalten.
- **Datenassoziationen**: Eingangs- und Ausgangsassoziationen verbinden Aktivitäten mit Datenobjekten und Datenspeichern und zeigen so Ein- und Ausgaben explizit an [18]. Sie sind ein wichtiges Signal für die Verarbeitung personenbezogener Daten, da sie den direkten Bezug einer Aktivität zu bestimmten Daten verdeutlichen (z.B. Lesezugriff auf eine Kundendatenbank).
- **Pools und Lanes**: Pools modellieren Organisationseinheiten oder Prozessbeteiligte, während Lanes Verantwortlichkeiten innerhalb eines Pools darstellen. Innerhalb eines Pools befinden sich die Aktivitäten und anderen Elemente des Prozesses [18]. Die Rollen und Verantwortlichkeiten, die durch Pools und Lanes dargestellt werden, können für die Bewertung der Datenverarbeitung relevant sein.
- **Nachrichtenflüsse**: stellen den Austausch von Nachrichten zwischen verschiedenen Pools dar [18]. Sie können auf eine Übermittlung personenbezogener Daten an Dritte hinweisen (z.B. Transfer von Kundendaten an einen externen Dienstleister).
- **Textannotationen und Assoziationen**: dienen dazu, zusätzliche Informationen zu Prozessmodellen hinzuzufügen, die nicht durch die standardmäßigen BPMN-Elemente abgedeckt sind [18]. Sie können genutzt werden, um die Art der Datenverarbeitung zu präzisieren (z.B. „enthält E-Mail-Adresse“).

BPMN-XML

BPMN-Modelle werden in einer XML-Serialisierung gespeichert (BPMN 2.0 XML) [18]. Diese Darstellung enthält alle relevanten Strukturinformationen (Elementtypen, Namen, Beziehungen, Zuordnungen, Positionen der Elemente) und wird von



(a) Ohne Datenassoziation (b) Mit Datenassoziation

Abbildung 2.1: Beispiel einer Datenassoziation als Datenschutzsignal.

vielen Prozess-Engines und Modellierungswerkzeugen wie Camunda [12] und BPMN.io [11] unterstützt. Für diese Arbeit dient BPMN-XML als Eingabeformat der Klassifizierungspipeline (siehe Kapitel 4).

Im Metamodell von BPMN erben fast alle Elemente von `BaseElement` und damit ein `id`-Attribut. Dieses `id` dient der eindeutigen Referenzierung und ist *erforderlich* [18]. Diese ID ist für die Klassifizierungspipeline wichtig, da sie eine stabile Referenzierung der Aktivitäten und anderer Elemente ermöglicht. Dies ist insbesondere dann relevant, wenn die Ergebnisse der Klassifizierung auf die ursprünglichen Prozessmodelle zurückgeführt werden müssen.

Beispiel einer Datenassoziation als Datenschutzsignal

Abbildung 2.1 zeigt ein einfaches Beispiel, wie eine Datenassoziation die DSGVO-Relevanz einer Aktivität verdeutlichen kann. In Abbildung 2.1a ist die Aktivität „Daten prüfen“ ohne Datenassoziation dargestellt, was wenig über die Art der verarbeiteten Daten aussagt. In Abbildung 2.1b hingegen zeigt die eingehende Datenassoziation von einem Datenspeicher „Kunden-DB“, dass die Aktivität personenbezogene Daten verarbeitet. Dies macht die Aktivität als potenziell datenschutzkritisch erkennbar. Dieses Beispiel unterstreicht die Notwendigkeit den gesamten Kontext einer Aktivität zu betrachten, um fundierte Rückschlüsse auf die Verarbeitung personenbezogener Daten ziehen zu können.

2.3 Large Language Models (LLMs)

LLMs sind große, vortrainierte Sprachmodelle, die auf der Transformer-Architektur basieren. Transformer, erstmals in [29] beschrieben, verarbeiten eine Eingabe nicht strikt sequenziell, sondern beachten alle Tokens einer Sequenz parallel. Über Self-Attention gewichten sie, welche Token füreinander relevant sind. Als Token gelten Wörter oder Wortbestandteile, in die der Text vorab zerlegt wird. Dieser Mechanismus erfasst Abhängigkeiten über große Distanzen innerhalb der Sequenz und ermöglicht dadurch eine effiziente Kontextmodellierung. Die Transformer-Architektur bildet heute das Fundament moderner Sprachmodelle wie der GPT-Familie von OpenAI [4, 16, 21].

In chatbasierten Systemen wird das Verhalten des LLM über System- und User-Prompts gesteuert. Gutes Prompt Engineering kann die Leistung und Format-Treue der Ausgabe verbessern, ohne dass die Modellparameter verändert werden müssen [15]. Ein deutlicher Vorteil aktueller LLMs ist Zero-/Few-Shot Learning. Damit lassen sich Aufgaben alleine über Instruktionen und wenige Beipsiele lösen, ohne dass erneutes Training benötigt wird [5, 15]. Das ist besonders nützlich für Klassifikationsaufgaben, bei denen nur wenige gelabelte Beispiele vorliegen, wie etwa die Identifikation von DSGVO-kritischen Aktivitäten in Prozessmodellen.

Um LLMs in automatisierten Pipelines zu integrieren sind schema-konforme Ausgaben, wie ein gültiges JSON, unerlässlich. In der Praxis gibt es dafür drei Ansätze:

1. Klare Angaben über das Ausgabeformat im System- oder User-Prompt [15].
2. API-gestützte Mechanismen wie Function Calling oder Structured-Output/JSON-Mode mit Schemaüberprüfung [3, 19, 22].
3. Constrained Decoding, das die Generierung auf eine vorgegebene Grammatik beschränkt. Ein Beispiel ist PICARD: Bei jedem Generationsschritt des Language Model (LM) werden nur zulässige Tokens ausgewählt [26].

Typische Fehlerbilder bei der Nutzung von LLMs sind Halluzinationen (plausibel wirkende, aber fehlerhafte Aussagen) und Formatfehler (wie z.B. ungültiges JSON). In [14] wird argumentiert, dass Halluzinationen bereits beim Erstellen des LLM durch die Trainings- und Evaluationsmethoden begünstigt werden, die das Modell dazu bringen, eher zu raten als Unsicherheit zuzugeben. Das Raten bei Unsicherheit

verbessert die Testergebnisse. Gegenmaßnahmen gegen Halluzinationen sind u.a. präzisere Prompts, Informationserweiterung des Prompts durch Retrieval Augmented Generation (RAG) und Self-Check/Retry-Strategien als Post-Processing Methoden nach der Generierung [13].

Die meisten großen LLMs werden von Unternehmen wie OpenAI, Google oder Anthropic entwickelt und als API-Dienste angeboten. In der Industrie zählt GPT-4o aktuell zu den am weit verbreitetsten Modellen [20]. Es ist ein multimodales Modell mit starken Text-, Bild- und Audiofähigkeiten. Properitäre Modelle wie GPT-4o sind oft leistungsstark, aber auch teuer und intransparent. Als Alternative gibt es eine wachsende Zahl an Open-Source LLMs, die frei verfügbar sind und lokal betrieben werden können. Beispiele sind Die von Mistral [2], Deepseek [1] und Qwen [24]. Open-Source Modelle bieten Vorteile wie Kostenersparnis, Datenschutz und Anpassbarkeit. In dieser Arbeit werden sowohl proprietäre als auch Open-Source LLMs evaluiert.

2.4 Verwandte Arbeiten

- Was für Ansätze gibt es bereits Prozesse automatisiert nach datenschutzkritischen Aktivitäten klassifizieren zu lassen
- LLMs zum Klassifizieren nutzen
- Prompt Engineering (Zero-Shot)
- Überblick über LLMs in Businessprozessen. Was gibt es bereits für Ansätze diese zu benutzen
- Benchmarking und Evaluierung von LLMs
- Identifizierte Forschungslücken: LLMs zur Klassifizierung, EU-Fokus, einheitliche reproduzierbare Benchmarks

3 Problemdefinition und Zielkriterien

3.1 Aufgabenstellung

- Eingabe ist BPMN-XML; Ausgabe ist eine Menge von Aktivitäts-IDs, die als kritisch klassifiziert worden sind. Optional auch mit Erklärung vom LLM warum es sich so entschieden hat
- Definition was als kritisch zu klassifizieren ist: Eine Aktivität ist kritisch, wenn sie personenbezogene Daten verarbeitet (Hier nochmal genauer definieren nach DSGVO), einschließlich Nutzung bereits vorhandener Daten.

3.2 Qualitätsziele

- Hauptziel ist ein hoher Recall mit minimalen False Negatives, bei noch akzeptabler Precision und somit auch überschaubaren False Positives

// TODO Aktuell unterstützt meine Evaluierung noch keine Seeds, aber das sollte ich noch umsetzen können und dann Temperature bei den LLMs auf 0 setzen

- Sekundäres Ziel ist Stabilität über mehrere Seeds hinweg
- Hier könnte ich noch bestimmte Wertebereiche vorschlagen, sie sinnvoll wären wie Recall $\geq 80\%$, False Positives $\leq 1{,}5/$Prozess. Unbedingt dafür noch in anderen Papern nach guten Werten schauen, wenn ich das machen sollte

3.3 Scope und Annahmen

// TODO Das muss ich noch anpassen

- Sprachen sind DE und EN
- Keine Bewertung der Rechtmäßigkeit an sich, sondern nur ein Screening nach kritischen Aktivitäten (Wie ein Vorfilter)
- Risiken und Grenzen (Bspw. schlechte Einschätzung des LLM wenn Artefakte wie Datenobjekte im BPMN-Modell fehlen) werden transparent gemacht

4 Klassifizierungsalgorithmus (Design und Implementierung)

- Im gesamten Kapitel werden die genutzten Technologien an geeigneter Stelle aufgezeigt und beschrieben (Camunda-BPMN, LangChain4j, Spring Boot, Kotlin, . . .)
  - TODO: Ggf. extra Unterkapitel für die genutzten Technologien oder doch immer da erwähnen wo es gerade passt
  - Aufzählungen für die Technologien nutzen

4.1 Ziel und Annahmen

- Eingabe ist BPMN-XML, Ausgabe sind die IDs der kritischen Aktivitäten mit ggf. Erklärung
- Getestet wird mit BPMN-Modellen aus Camunda und bpmn.io
- Der Algorithmus klassifiziert Aktivitäten in BPMN-Prozessen binär nach “kritisch” oder “nicht kritisch”
- Ausgegeben werden ausschließlich die IDs der kritischen Aktivitäten ggf. mit Erklärung, warum sie als kritisch klassifiziert wurde
- Das Ziel ist ein robustes und reproduzierbares Verhalten über unterschiedliche Modelle
  - Granularität der Prozesse (Bspw. $>$5 Aktivitäten, 20+ Aktivitäten)
  - Gateways, Datenobjekte, Datenbanken

  - Mehrere Pools/Lanes, Message Flows
  - Evtl. (?) verschiedene Sprachen, Text-Annotationen

4.2 BPMN Preprocessing

- BPMN-XML wird geparst
- Relationen von Flow-Elementen zu anderen Elementen werden extrahiert:

```
data class BpmnElement(
    val type: String,
    val id: String,
    val documentation: String? = null,
    val name: String? = null,
    val outgoingFlowElementIds: List<String> = emptyList(),
    val incomingFlowElementIds: List<String> = emptyList(),
    val incomingDataFromElementIds: List<String> = emptyList(),
    val outgoingDataToElementIds: List<String> = emptyList(),
    val associatedElementIds: List<String> = emptyList()
)
```

- Dadurch entsteht für jedes Element ein strukturierter Kontext (id, name, pool, lane, eingehende Elemente, ...)
- Dieser Kontext wird später im Prompt genutzt, um dem LLM alle notwendigen Informationen strukturiert bereitzustellen

4.3 Prompt Engineering

- Referenz auf Zero-Shot/Few-Shot (?)
- Prompt-Sprache (Passendes Paper referenzieren)

- Erklärung des System-Prompt (Anleitung was als kritisch klassifiziert werden soll, Auflistung wichtiger DSGVO-kritischer Inhalte und Definitionen aus der DSGVO wie “Verarbeitung”, Definition des Ausgabeformats mit LangChain4j) (System-Prompt im Anhang beifügen oder hier direkt integrieren)
- Was steht im User-Prompt drin

4.4 Validierung der Ausgabe

- Ausgabeschema wird in LangChain4j deklarativ beschrieben
- Erklären was LangChain4j im Hintergrund tut, damit das Schema eingehalten wird und die Ausgabe das korrekte Format hat

```
data class AnalysisResponse(
    val criticalElements: List<CriticalElement>
) {
    data class CriticalElement(
        val id: String,
        val name: String?,
        val reason: String
    )
}
```

- Abgleichen der ausgegebenen IDs mit den vorhandenen aus dem Prozess. ggf. unzulässige entfernen

// TODO Entscheiden, ob ich hier noch Ablationen brauche (Falls ja hier noch ein Kapitel darüber einbauen) wie eine Baseline ohne zusätzliches Prompt Engineering, Varianten wo beim Preprocessing Lanes und Pools, Datenobjekte etc. weggelassen werden

4.5 API Design

- Klassifizierungspipeline ist über HTTP-Endpunkt aufrufbar, wo das BPMN-XML und ggf. Attribute zum Überschreiben des genutzten LLMs übergeben

werden

- Klassifizierungspipeline kann außerdem lokal über CLI gestartet werden und legt Ergebnisse in Datei ab
- Dadurch kann die Klassifizierung leicht in das Evaluationsframework eingebunden werden, während das Evaluationsframework flexibel bleibt und beliebige Klassifizierungsalgorithmen benutzen kann, welche ebenfalls die gleiche HTTP-Schnittstelle anbieten
- Beispielaufruf des Klassifizierungsendpunkts
- Irgendwo hier oder auch in einem nächsten Abschnitt die “Schnittstelle” mit BPMN-XML rein und JSON mit Liste der kritischen Elemente (evtl. mit Begründung) wieder. Durch diese Vereinheitlichung ist es möglich verschiedene Algorithmen in dem Evaluationsframework zu vergleichen (Stichwort Standardisierung). Vielleicht gehe ich auch soweit, dass ich den Standard für zukünftige Arbeiten propose, damit spätere Arbeiten meinen Standard zum Vergleich nutzen können

4.6 Nutzung über Frontend

- Die Klassifizierung kann über eine Sandbox im Frontend genutzt werden. Dort können auch die LLM-Parameter angepasst werden, um andere Modelle testen zu können
- Sandbox ist ein voller BPMN-Editor basierend auf BPMN.js und bietet zusätzlich die Funktionalität zur Klassifizierung an
- Als kritisch klassifizierte Elemente werden im Editor farblich hervorgehoben

5 Evaluationsframework

5.1 Anforderungen und Use-Cases

- Das Framework ermöglicht einen Vergleich von Modellen (Austauschbar durch LLMPropsOverride) und Algorithmen (Austauschbar über HTTP-Endpunkt) auf Basis von gelabelten Testdaten
- Evaluations-Run über YAML konfigurierbar (Modelle, Endpunkte, Testdaten)
- Detaillierte Ausgabe der Ergebnisse
  - Side-by-side Diagramme von Accuracy, Precision, Recall und F1-Score sowie FP, FN, TP, TN und generell wie viele Testcases erfolgreich waren
  - Detailliertere Ansicht nochmal pro Modell und dafür nochmal pro Testcase
- Zielnutzer sind Forschende und Entwickler die Modelle und Algorithmen auswerten und ggf. untereinander vergleichen wollen
- Frontend zur einfachen Bedienung

5.2 Architektur und Komponenten

- Diagramm zur Architektur erstellen
- HTTP-Endpunkt oder CLI zum Starten -> Holen der notwendigen Testdatensätze -> Aufruf der Klassifizierung pro Modell und pro Testcase -> Akkumulieren der Ergebnisse pro Testcase + ggf. frühzeitige Rückgabe des Ergebnisses des Testcase für Live-Ansicht in UI -> Aufarbeiten der akkumulierten Ergebnisse und berechnen wichtiger Metriken

- EvaluationController, MultiEvaluationRunner, EvaluationRunner, HttpEvaluator, MetricsAccumulator, ggf. Frontend (Concurrency von MultiEvaluationRunner)

5.3 Konfiguration einer Evaluierung

- Evaluierung kann mit einer YAML Datei konfiguriert werden

```
defaultEvaluationEndpoint: /gdpr/analysis/prompt-engineering
maxConcurrent: 10
models:
  - label: Mistral Medium 3.1
    llmProps:
      baseUrl: https://openrouter.ai/api/v1
      modelName: mistralai/mistral-medium-3.1
      apiKey: ${OPEN_ROUTER_API_KEY}
  - label: Deepseek Chat v3.1
    llmProps:
      baseUrl: https://openrouter.ai/api/v1
      modelName: deepseek/deepseek-chat-v3.1
      apiKey: ${OPEN_ROUTER_API_KEY}
  - label: GPT oss 120b
    llmProps:
      baseUrl: https://openrouter.ai/api/v1
      modelName: openai/gpt-oss-120b
      apiKey: ${OPEN_ROUTER_API_KEY}
datasets:
  - 2
  - 7
```

- Secrets können entweder im Klartext angegeben werden oder sicher mit `${...}` referenziert werden, wenn sie im Backend als Umgebungsvariable hinterlegt sind

5.4 Testdaten

- Die Testdaten werden aus einer Datenbank ausgelesen
- In der Konfig kann konfiguriert werden welche Testdaten genutzt werden sollen
- Die Testdaten können direkt in der App erstellt, verwaltet und gelabelt werden
- (Kapitel 6 wird sich um die Labelingsoftware drehen)

5.5 Generierte Resultate

- Für jeden Testfall werden pro Modell Ergebnisse gesammelt (klassifizierte Aktivitäten mit Erklärung, TP/FP/FN/TN, Bild-URL mit farblich hervorgehobenen Aktivitäten, bestanden oder nicht bestanden)
- Pro Modell werden "Summary-Objekte" erstellt mit jeweils Precision, Accuracy, Recall, F1-Score, Confusion (TP/FP/FN/TN), Anzahl korrekter Testfälle, Anzahl falsch klassifizierter Testfälle und Anzahl Testfälle in denen es zu Problemen kam (Bspw. Parsing Fehler, Token Limit überschritten, . . .)
- Generelle Metadaten über die genutzten Modelle, Endpunkte zur Klassifizierung und Testdatensätze, sowie Zeitstempel (und Seed????)

5.6 Visualisierung im Frontend

- Detaillierte Ausgabe der Ergebnisse
  - Side-by-side Diagramme von Accuracy, Precision, Recall und F1-Score sowie FP, FN, TP, TN und generell wie viele Testcases erfolgreich waren
  - Detailliertere Ansicht nochmal pro Modell und dafür nochmal pro Testcase
- Hier auch ruhig Bilder vom Frontend einbauen

- Die Ergebnisse können als JSON exportiert und wieder importiert werden und als Markdown Report exportiert werden

5.7 Erweiterbarkeit

- Neue Modelle können über Konfiguration ergänzt werden (Endpunkt, Modellname, API Key)
- Neue Klassifizierungsalgorithmen/pipelines können ergänzt werden, wenn sie das vorgegebene HTTP-Schnittstelle unterstützen
- Die Testdatensätze werden aus Datenbank ausgelesen und können für jeden Evaluations-Run neu gesetzt werden

6 Labeling und Datensätze

6.1 Labeling Tool

Hier brauche noch genaue Kapitel, aber hier soll auf jeden Fall folgendes rein:

- Definition von Labeln hier, oder kommt das schon vorher? (Muss im Verlauf beim schreiben entschieden werden)
- Labeling Software wurde entwickelt
- Es können Datensätze mit Namen und Beschreibungen erstellt werden
- Für jeden Datensatz können beliebig viele Testcases erstellt werden
- Ein Testcase ist ein BPMN Diagramm, welches direkt in der App mithilfe von bpmn.io bearbeitet werden kann
- Zusätzlich bietet der Editor eine Labeling Funktionalität, mit welcher Aktivitäten, welche als kritisch erkannt werden sollen, gelabelt werden. Zusätzlich kann auch eine Erklärung angegeben werden
- Datensätze und gelabelte Testcases werden in Datenbank gespeichert und werden während der Evaluierung des Evaluationsframeworks benutzt

6.2 Quellen und Eigenschaften der Datensätze

- Es werden drei unterschiedliche Datensätze genutzt
  - Von der Uni bekommen
  - Mittelgroße realistische Prozesse aus unterschiedlichen Branchen (mit Pools, Lanes, Datenobjekten, Gateways, . . .)

  - Kleine Prozesse mit maximal 5 Aktivitäten und keinen weiteren Elementen

- Eventuell tabellarische Aufzählung von Eckdaten zu jedem Datensatz (Anzahl Elemente, Sprache, benutzte Elemente)
- Hier sollte ich glaube ich noch besser erklären warum die Auswahl heterogen ist (und damit möglichst aussagekräftige Ergebnisse in der Evaluierung erzeugen)
- Eventuell (im Anhang) eine Liste aller Testfälle jeweils auch mit den Eckdaten und auf diese verweisen. Da kann dann auch immer noch eine kurze Beschreibung dazu was daran besonders ist

6.3 Labeling-Guide

- Eine Aktivität ist als kritisch gelabelt, wenn sie personenbezogene Daten erhebt, nutzt, speichert, übermittelt oder löscht oder anderweitig explizit eine DSGVO-Pflicht auslöst (Auskunft, Löschung) (Hier auch die Kriterien von der originalen DSGVO einbinden und daran ableiten, wie man labeln soll)
- Beispiele einbauen, was als kritisch gelabelt wurde und Gegenbeispiele aufzeigen, damit Grenzfälle klar definiert sind

7 Modellauswahl

7.1 Kriterien

- Ab wann gilt ein Modell als EU-Modell, entsprechend umgekehrt: Ab wann ist ein Modell international
  - Sitz in EU
  - Training/Fine-Tuning in EU
  - Veröffentlicht in EU
- Was bedeutet Open Source
  - Frei verfügbare Gewichte
  - Permissive Lizenz
- Größenklassen (Bspw. $<$8B, 8–20B, . . .)
- Herkunftsland, Hosting-Land
- Letztes Update
- Downloads
- Lizenz
- Kontext (Wichtig auch für die Größe der Prozesse, die damit verarbeitet werden können. Irgendwo muss ich das auch nochmal thematisieren)
- Ich brauche noch irgendwo den Stand/das Datum nach dem ich nicht mehr nach neuen Modellen gesucht habe

7.2 Modellvorstellung

- Tabellarische Auflistung der Modelle
- Begründung warum genau die Modelle ausgewählt worden sind

8 Versuchsaufbau

Ich habe noch nicht die Unterkapitel (außer Metriken) aber folgendes soll hier u. a. rein:

- Die Modelle werden jeweils mit dem gleichen Klassifizierungsalgorithmus und den gleichen Datensätzen benutzt
- Die Evaluierung wird jeweils für jeden vorhandenen Testdatensatz durchgeführt (Uni, reale größere Prozesse, kleine Prozesse)
- Die Evaluationen werden pro Konfiguration (Modelle, Datensätze) mehrfach durchgeführt (Unterschiedliche Seeds, falls ich bis dahin Seeds unterstütze)
- Es werden verschiedene LLM-Modelle vergleichen
- Es werden vom gleichen LLM-Modell die unterschiedlichen Größen verglichen
- ...
- Ein Testcase gilt als korrekt klassifiziert, wenn genau die als kritisch gelabelten Aktivitäten als kritisch klassifiziert worden sind. Sobald es False Positives oder False Negatives gibt, ist ein Testcase nicht korrekt klassifiziert worden
- LLM Temperature 0 vorstellen (Falls ich das mit den Seeds noch umsetze) für reproduzierbare Ergebnisse

8.1 Metriken

(Dopplung mit 5.5. ist das schlimm?)

- Für jeden Testfall werden pro Modell Ergebnisse gesammelt (klassifizierte Aktivitäten mit Erklärung, TP/FP/FN/TN, Bild-URL mit farblich hervorgehobenen Aktivitäten, bestanden oder nicht bestanden)
- Pro Modell werden “Summary-Objekte” erstellt mit jeweils Precision, Accuracy, Recall, F1-Score, Confusion (TP/FP/FN/TN), Anzahl korrekter Testfälle, Anzahl falsch klassifizierter Testfälle und Anzahl Testfälle in denen es zu Problemen kam (Bspw. Parsing Fehler, Token Limit überschritten, . . .)

9 Durchführung

9.1 Experimente

- Dokumentation der einzelnen Runs
  - Konfiguration aufzeigen
  - Diagramme der Ergebnisse

9.2 Fehlerklassen

- Falls es Fehlerklassen gibt kann ich sie hier thematisieren (Timeout, Parse-Error, Token Limit, . . .)

10 Ergebnisse

10.1 Gesamtübersicht

- Hier allgemein die Ergebnisse der einzelnen Runs darstellen (Vor allem die Diagramme wo die einzelnen Metriken nebeneinander aufgelistet sind)

10.2 Analyse

- Aufschlüsselung nach Datensätzen
- Aufzeigen sichtbarer Muster

10.3 Antworten auf Forschungsfragen

- Hier werden die unteren Forschungsfragen mithilfe der Evaluationsergebnisse beantwortet
- EU vs. International
- Groß vs. Klein
- Bestes Open-Source-Modell
- Open Source vs. Kommerziell

10.4 Fallstudien

- Hier werde ich (wahrscheinlich 2–3) spezifische exemplarische Ergebnisse von Testcases heraussuchen und genauer unter die Lupe nehmen. Was ist hier besonders aufgefallen oder interessant?
- Hier kann ich die Bilder mit den Highlights hernehmen
- Idealerweise habe ich Beispiele für 1. TN, 2. FP aber plausible Erklärung und 3. FN

10.5 Robustheit

- Varianz über mehrere Runs untersuchen (Unterschiedliche Seeds, falls ich bis dahin welche unterstütze)
- Ggf. weitere Metriken hier interessant
- Wie fehleranfällig ist die Klassifizierung

11 Diskussion

11.1 Interpretation der Befunde

- Einordnung der Rangfolge der LLMs
- Besonderheiten der Modellfamilien (Bspw. wie groß ist der Unterschied von Groß gegen Klein?)
- Es gab Prozesse in denen ich eine Aktivität nicht als kritisch gelabelt habe, aber das LLM in der Evaluierung das als kritisch mit einer validen Begründung klassifiziert hat, man könnte die Testdaten also noch anpassen, wenn die Begründung des LLMs überzeugt (Ich weiß nicht wie sinnvoll das ist hier zu thematisieren) -> hier eher in die Richtung gehen, dass lieber mehrere Personen labeln als auf die Anpassung der Datensätze zu gehen.

11.2 Hoher Recall vs. Präzision

- Beobachtung ausformulieren, dass einige Testfälle als fehlerhaft eingeordnet wurden, weil es False-Positives gab, obwohl es keine False-Negatives gab. Es wurden also alle Aktivitäten gefunden, die gefunden werden sollten, nur halt noch mehr on top.
- Einordnung der FP-Last pro Prozess (Lieber False Positives, als dass etwas übersehen wird, Ziel war sowieso ein Vorscreening), Diskussion darüber wie nützlich hohe Recall Werte sind

11.3 EU-Modelle

- Analyse der EU-Open-Source-Modelle in Bezug auf Precision, Recall und Stabilität in Bezug auf die anderen Modelle
- Wie gut haben sich die EU Modelle im Vergleich zu den anderen geschlagen

11.4 Open-Source Modelle

- Analyse der Open-Source-Modelle in Bezug auf Precision, Recall und Stabilität in Bezug auf kommerzielle Modelle
- Wie gut haben sich die Open-Source-Modelle im Vergleich zu den anderen geschlagen

11.5 Modellgrößen

- Selbst hosten von Modellen diskutieren. Ist es realistisch die Modelle selbst zu hosten, welche gut performt haben? Reichen die kleinen Varianten der Modelle oder muss man schon die großen Modelle benutzen, um gute Ergebnisse zu erzielen

11.6 Grenzen

- Wären Grenzen wie BPMN-Modellgröße im Zusammenhang mit der Kontextlänge des LLM interessant?
- Keine aussagekräftige Rechtsberatung, sondern stand jetzt eher ein Vorscreening, was nochmal überprüft werden muss
- ggf. notwendige Anonymisierung von Prozessen diskutieren (Wenn das in BPMN Modellen überhaupt ein Problem ist)

12 Zusammenfassung

Hier neben der allgemeinen Zusammenfassung unbedingt noch die erste Forschungsfrage explizit beantworten

13 Aussicht

Unter anderem das hier, evtl noch mehr:

- Jetzt gibt es ein einheitliches Evaluationsframework mit einer einheitlich definierten Schnittstelle für Klassifizierungsalgorithmen -> Zukünftige Arbeiten können sich mit der Entwicklung besserer Klassifizierungsalgorithmen/Pipelines (Bspw. noch RAG einbauen) beschäftigen und diese mit diesem Framework vergleichen/benchmarken
- Außerdem können in Zukunft auch noch mehr Modelle vergleichen werden, da sich die Welt der LLMs rasant weiterentwickelt
- Auch Finetunen ist etwas was interessant gewesen wäre für diese Masterarbeit, aber den Rahmen gesprengt hätte

A Quelltexte

In diesem Anhang sind einige wichtige Quelltexte aufgeführt.

```
1 public class Hello {
2     public static void main(String[] args) {
3         System.out.println("Hello World");
4     }
5 }
```

Literatur

[1] DeepSeek AI. *DeepSeek AI Open Source Hugging Face Models*. 2025. URL: https://huggingface.co/deepseek-ai (besucht am 17. 07. 2025).

[2] Mistral AI. *Mistral AI*. 2025. URL: https://mistral.ai/ (besucht am 21. 09. 2025).

[3] Mistral AI. *Mistral AI - Structured Output*. 2025. URL: https://docs.mistral.ai/capabilities/structured-output/structured_output_overview/ (besucht am 11. 07. 2025).

[4] Ivan Belcic und Cole Stryker. *Was ist ein GPT (Generative Pre-Trained Transformer)?* Sep. 2024. URL: https://www.ibm.com/de-de/think/topics/gpt (besucht am 18. 09. 2025).

[5] Tom Brown u. a. „Language Models are Few-Shot Learners“. In: *Advances in Neural Information Processing Systems*. Hrsg. von H. Larochelle u. a. Bd. 33. Curran Associates, Inc., 2020, S. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[6] Bundesministerium der Justiz. *Gesetz über außergerichtliche Rechtsdienstleistungen (Rechtsdienstleistungsgesetz - RDG)*. https://www.gesetze-im-internet.de/rdg/. Dez. 2007. (Besucht am 15. 08. 2025).

[7] Antonio Capodieci u. a. „BPMN-Enabled Data Protection and GDPR Compliance“. In: *IS-EUD Workshops*. 2023. URL: https://api.semanticscholar.org/CorpusID:259099646.

[8] Giovanni Ciaramella u. a. „Leveraging Pre-trained LLMs for GDPR Compliance in Online Privacy Policies“. In: (2022). URL: https://ceur-ws.org/Vol-3962/paper44.pdf.

[9] Europäische Union. *Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates vom 27. April 2016 zum Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten, zum freien Datenverkehr und zur Aufhebung der Richtlinie 95/46/EG (Datenschutz-Grundverordnung)*. https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32016R0679. 2016.

[10] European Data Protection Board. *Guidelines 4/2019 on Article 25 Data Protection by Design and by Default*. https://www.edpb.europa.eu/sites/default/files/files/file1/edpb_guidelines_201904_dataprotection_by_design_and_by_default_v2.0_en.pdf. Version 2.0. Okt. 2020.

[11] Camunda Services GmbH. *BPMN.io - Web-based tooling for BPMN, DMN and Forms*. 2025. URL: https://bpmn.io/ (besucht am 22. 09. 2025).

[12] Camunda Services GmbH. *Camunda Platform*. 2025. URL: https://camunda.com/de/ (besucht am 22. 09. 2025).

[13] Ziwei Ji u. a. „Survey of Hallucination in Natural Language Generation“. In: *ACM Comput. Surv.* 55.12 (März 2023). ISSN: 0360-0300. DOI: 10.1145/3571730. URL: https://doi.org/10.1145/3571730.

[14] Adam Tauman Kalai u. a. *Why Language Models Hallucinate*. 2025. arXiv: 2509.04664 [cs.CL]. URL: https://arxiv.org/abs/2509.04664.

[15] Pengfei Liu u. a. „Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing“. In: *ACM Comput. Surv.* 55.9 (Jan. 2023). ISSN: 0360-0300. DOI: 10.1145/3560815. URL: https://doi.org/10.1145/3560815.

[16] Shervin Minaee u. a. *Large Language Models: A Survey*. 2025. arXiv: 2402.06196 [cs.CL]. URL: https://arxiv.org/abs/2402.06196.

[17] Leonard Nake u. a. „Towards identifying gdpr-critical tasks in textual business process descriptions“. In: (2023). URL: https://dl.gi.de/server/api/core/bitstreams/84ac5110-1a0f-4e3c-bdf8-6393555a7212/content.

[18] OMG. *Business Process Model and Notation (BPMN)*. 2011. URL: https://www.omg.org/spec/BPMN/2.0/PDF (besucht am 01. 05. 2023).

[19] OpenAI. *Function calling and other API updates*. https://openai.com/index/function-calling-and-other-api-updates/. 2023. (Besucht am 10. 07. 2025).

[20] OpenAI. *Hello GPT-4o*. Mai 2024. URL: https://openai.com/index/hello-gpt-4o/ (besucht am 21. 07. 2025).

[21] OpenAI. *Model Overview*. 2025. URL: https://platform.openai.com/docs/models (besucht am 18. 09. 2025).

[22] OpenAI. *OpenAI - Structured model outputs*. URL: https://docs.mistral.ai/capabilities/structured-output/structured_output_overview/ (besucht am 11. 07. 2025).

[23] KC Pragyan u. a. „Toward Regulatory Compliance: A few-shot Learning Approach to Extract Processing Activities“. In: *2024 IEEE 32nd International Requirements Engineering Conference Workshops (REW)*. IEEE. 2024, S. 241–250. URL: https://ieeexplore.ieee.org/abstract/document/10628578.

[24] Alibaba Qwen. *Qwen Open Source Hugging Face Models*. 2025. URL: https://huggingface.co/Qwen (besucht am 17. 07. 2025).

[25] Konrad Schneid u. a. „Uncovering data-flow anomalies in BPMN-based process-driven applications“. In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. 2021, S. 1504–1512. URL: https://dl.acm.org/doi/abs/10.1145/3412841.3442025.

[26] Torsten Scholak, Nathan Schucher und Dzmitry Bahdanau. „PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models“. In: *CoRR* abs/2109.05093 (2021). arXiv: 2109.05093. URL: https://arxiv.org/abs/2109.05093.

[27] Magdalena von Schwerin und Manfred Reichert. „A systematic comparison between open-and closed-source large language models in the context of generating gdpr-compliant data categories for processing activity records“. In: *Future Internet* 16.12 (2024), S. 459. URL: https://www.mdpi.com/1999-5903/16/12/459.

[28] Ángel Jesús Varela-Vaca u. a. „Business process models and simulation to enable GDPR compliance“. In: *International Journal of Information Security* 24.1 (2025), S. 41. URL: https://link.springer.com/article/10.1007/s10207-024-00952-7.

[29] Ashish Vaswani u. a. „Attention Is All You Need“. In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

Name: Merten Dieckmann Matrikelnummer: 1058340

Erklärung

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Merten Dieckmann