



Technische  
Hochschule  
Nürnberg

Fakultät Informatik

**Vergleichsanalyse von multimodalen Large  
Language Models und einer  
OCR/YOLO-Pipeline zur  
Dokumentenklassifikation für die  
Bundesagentur für Arbeit**

Bachelorarbeit im Studiengang Informatik

vorgelegt von

Lukas Müller

Matrikelnummer 3698673

Erstgutachter: Prof. Dr. Natalie Kiesler

Zweitgutachter: Prof. Dr. Korbinian Riedhammer

© 2026

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.

## Kurzdarstellung

Angesichts des demografischen Wandels und steigender Antragszahlen ist die Automatisierung von Verwaltungsprozessen für die Bundesagentur für Arbeit unverzichtbar. Das bestehende System zur teilautomatisierten Verarbeitung von Ausbildungsnachweisen, basierend auf einer OCR/YOLO-Pipeline, weist jedoch Defizite bei variablen Dokumentenlayouts auf.

Ziel dieser Arbeit ist die Evaluation multimodaler Large Language Models als potenzielles Ersatzsystem. Verglichen wurden die Modelle Pixtral-12B, Qwen-2.5-VL-7B und Qwen-2.5-VL-32B hinsichtlich ihrer Klassifikations- und Extraktionsleistung sowie ihrer Ressourceneffizienz (Latenz, Energie, VRAM). Ein Schwerpunkt lag auf der Untersuchung, ob ein domänen spezifisch nachtrainiertes 7B-Modell mit leistungsstärkeren Basismodellen konkurrieren kann.

Die Ergebnisse belegen, dass die bisherige Pipeline bei komplexen Verträgen lediglich einen niedrigen F1-Score aufweist. Das Qwen-2.5-VL-32B liefert die qualitativ besten Ergebnisse, ist jedoch aufgrund der Ressourceneffizienz ungeeignet. Als Handlungsempfehlung wird der Einsatz des nachtrainierten Qwen-2.5-VL-7B ausgesprochen. Mit der Leistungssteigerung bietet dieses Modell den besten Kompromiss zwischen Leistungsfähigkeit und Ressourceneffizienz.



# Inhaltsverzeichnis

<b>1 Einleitung</b>	.	.	.	.	.	.	.	.	.	.	.	.	<b>1</b>
1.1 Motivation	.	.	.	.	.	.	.	.	.	.	.	.	<b>1</b>
1.2 Problemstellung	.	.	.	.	.	.	.	.	.	.	.	.	<b>1</b>
1.3 Zielsetzung	.	.	.	.	.	.	.	.	.	.	.	.	<b>2</b>
<b>2 Theoretische Grundlagen</b>	.	.	.	.	.	.	.	.	.	.	.	.	<b>3</b>
2.1 Dokumentenarten	.	.	.	.	.	.	.	.	.	.	.	.	<b>3</b>
2.1.1 KG5b-Formular	.	.	.	.	.	.	.	.	.	.	.	.	<b>3</b>
2.1.2 Ausbildungsvertrag	.	.	.	.	.	.	.	.	.	.	.	.	<b>3</b>
2.1.3 Sonstige Dokumente	.	.	.	.	.	.	.	.	.	.	.	.	<b>4</b>
2.2 Fachliche Grundlagen	.	.	.	.	.	.	.	.	.	.	.	.	<b>4</b>
2.2.1 Der Ist-Zustand	.	.	.	.	.	.	.	.	.	.	.	.	<b>4</b>
2.2.2 Rahmenbedingungen und Infrastruktur	.	.	.	.	.	.	.	.	.	.	.	.	<b>6</b>
2.3 Vision Language Models (VLMs)	.	.	.	.	.	.	.	.	.	.	.	.	<b>7</b>
2.4 Vorstellung der Modelle	.	.	.	.	.	.	.	.	.	.	.	.	<b>8</b>
2.4.1 Pixtral-12B	.	.	.	.	.	.	.	.	.	.	.	.	<b>9</b>
2.4.2 Qwen-2.5-VL	.	.	.	.	.	.	.	.	.	.	.	.	<b>9</b>
2.4.3 Benchmarks	.	.	.	.	.	.	.	.	.	.	.	.	<b>9</b>
2.5 Parameter-Efficient Fine-Tuning	.	.	.	.	.	.	.	.	.	.	.	.	<b>9</b>
2.6 Verwandte Arbeiten	.	.	.	.	.	.	.	.	.	.	.	.	<b>10</b>
2.6.1 Die Grenzen sequentieller OCR-Pipelines	.	.	.	.	.	.	.	.	.	.	.	.	<b>10</b>
2.6.2 Semantische Layout-Modellierung	.	.	.	.	.	.	.	.	.	.	.	.	<b>11</b>
2.6.3 End-To-End-Architekturen und multimodale Ansätze	.	.	.	.	.	.	.	.	.	.	.	.	<b>11</b>
<b>3 Methodik und Implementierung</b>	.	.	.	.	.	.	.	.	.	.	.	.	<b>13</b>
3.1 Modellierung des Dokumentenbestands als Grundlage der Evaluation	.	.	.	.	.	.	.	.	.	.	.	.	<b>14</b>
3.1.1 Preprocessing des Dokumentenbestands	.	.	.	.	.	.	.	.	.	.	.	.	<b>14</b>
3.1.2 Modellierung der Dokumentenarten als JSON-Objekte	.	.	.	.	.	.	.	.	.	.	.	.	<b>15</b>
3.1.3 Test- und Validierungsdatensatz	.	.	.	.	.	.	.	.	.	.	.	.	<b>17</b>
3.2 Evaluation der generierten JSON-Objekte	.	.	.	.	.	.	.	.	.	.	.	.	<b>17</b>
3.2.1 Validierung des VLM-Outputs	.	.	.	.	.	.	.	.	.	.	.	.	<b>17</b>
3.2.2 Vergleichslogik der unterschiedlichen JSON-Felder	.	.	.	.	.	.	.	.	.	.	.	.	<b>17</b>
3.2.3 Bewertung der Klassifikation und der Information Extraction	.	.	.	.	.	.	.	.	.	.	.	.	<b>19</b>
3.3 Messung von Latenz, Energieverbrauch und VRAM-Auslastung	.	.	.	.	.	.	.	.	.	.	.	.	<b>21</b>

3.4 Auswahl des Basismodells . . . . .	22
3.5 Fine-Tuning des Basismodells . . . . .	23
3.5.1 Trainingsdatensatz . . . . .	23
3.5.2 Optimierung des Fine-Tunings . . . . .	24
<b>4 Ergebnisse . . . . .</b>	<b>27</b>
4.1 Ergebnisse der OCR/YOLO-Pipeline gegenüber den Basismodellen . . . . .	27
4.2 Ergebnisse des weitertrainierten Modells gegenüber dem größeren Basismodell . .	31
4.3 Ressourcenverbrauch der VLMs gegenüber der OCR/YOLO-Pipeline . . . . .	34
<b>5 Diskussion . . . . .</b>	<b>37</b>
5.1 Interpretation der Fehler . . . . .	37
5.2 Abwägung Effektivität und Ressourceneffizienz . . . . .	38
5.3 Komplexität und Wartbarkeit der Systemarchitekturen . . . . .	39
5.4 Zusammenfassende Beantwortung der Forschungsfragen . . . . .	40
5.4.1 Beantwortung der Forschungsfrage 1 . . . . .	40
5.4.2 Beantwortung der Forschungsfrage 2 . . . . .	41
5.4.3 Beantwortung der Forschungsfrage 3 . . . . .	42
5.5 Limitationen und Probleme der Evaluation . . . . .	43
5.6 Handlungsempfehlung für die Bundesagentur für Arbeit . . . . .	45
<b>6 Zusammenfassung und Ausblick . . . . .</b>	<b>47</b>
6.1 Zusammenfassung . . . . .	47
6.2 Ausblick . . . . .	48
<b>Abbildungsverzeichnis . . . . .</b>	<b>49</b>
<b>Tabellenverzeichnis . . . . .</b>	<b>51</b>
<b>Listingverzeichnis . . . . .</b>	<b>53</b>
<b>Literaturverzeichnis . . . . .</b>	<b>55</b>
<b>Glossar . . . . .</b>	<b>57</b>

# Kapitel 1

## Einleitung

### 1.1 Motivation

Die öffentliche Verwaltung in Deutschland steht vor einer großen Herausforderung. Durch den demografischen Wandel verliert der öffentliche Sektor eine hohe Anzahl an erfahrenen Sachbearbeitern [Arbe25], während der Anspruch der Bürger steigt. Dieser Wandel wird ohne Digitalisierung und Automatisierung der Prozesse kaum abzufangen sein.

Auch in der Bundesagentur für Arbeit (BA) macht sich dieser Wandel hin zu mehr Digitalisierung bemerkbar. Ein Beispiel hierfür ist die Bearbeitung von Kindergeldanträgen. Mit einem jährlichen Aufkommen von mehreren Millionen Anträgen [Arbe24] und saisonalen Spitzen kommen hier die Sachbearbeiter an ihre Grenzen.

Um sich dieser Herausforderung zu stellen, startete bereits im Oktober 2025 der Produktivbetrieb eines teilautomatisierten Systems. Der Fokus liegt hierbei auf den Kindergeldanträgen volljähriger Auszubildender. Das System klassifiziert die Dokumente, erkennt mithilfe von YOLO- und OCR-Modellen Inhalte aus hochgeladenen Dokumenten und zeigt dem Sachbearbeiter entsprechende Vorschläge an. Diese aktuelle OCR/YOLO-Pipeline wertet die Bearbeitung der Kindergeldanträge für die Sachbearbeiter bereits auf, stößt jedoch an ihre Grenzen.

Aufgrund der schnellen Entwicklung im Bereich der Vision Language Models (VLMs) werden diese zunehmend relevanter für Aufgaben im Bereich der Dokumentenverarbeitung [Li25]. Die vorliegende Arbeit evaluiert, inwiefern ein solches Modell das Potenzial besitzt, die OCR/YOLO-Pipeline zu ersetzen.

### 1.2 Problemstellung

Seitdem die aktuelle Pipeline im operativen Einsatz ist, zeigen sich verschiedene Herausforderungen.

Das Hauptproblem ist die hohe Varianz der hochgeladenen Dokumente, insbesondere der Ausbildungsverträge. Die verschiedenen Firmen und Kammern nutzen alle unterschiedliche Layouts, wodurch das Training der Modelle komplex ist. Es ist schwierig, die Varianz in den Trainingsdaten abzubilden, zumal es zeitaufwendig ist, eine solche Menge an Trainingsdaten bereitzustellen.

Die Dokumente, die relevant für den Kindergeldantrag bei volljährigen Auszubildenden sind, benötigen in allen Fällen eine Unterschrift. Dadurch werden die Dokumente zwingenderweise ausgedruckt, ausgefüllt und unterschrieben, um schlussendlich wieder eingescannt zu werden. Dieser Prozess, auch Medienbruch genannt, hat eine schlechte Qualität zur Folge.

Angesichts der Verwendung von vielen unterschiedlichen Modellen ist die Wartung des Systems aufwendig. Eine Anpassung an neue Gegebenheiten ist ein zeitaufwendiger Prozess. Schon kleine Änderungen an der Laufzeitumgebung, wie zum Beispiel eine neue Python-Version, können ein erneutes Training erforderlich machen.

### 1.3 Zielsetzung

Das Ziel dieser Bachelorarbeit ist die Entwicklung und Evaluation eines prototypischen Systems zur teilautomatisierten Klassifikation und Informationsextraktion von Dokumenten. Angesichts der hohen Anzahl manuell bearbeiteter Anträge soll untersucht werden, inwie weit VLMs diesen Prozess effizienter gestalten können.

Die Arbeit umfasst die Implementierung einer Pipeline, die gescannte Dokumente als Bilddaten verarbeitet. Das System soll den Dokumententyp eigenständig erkennen und definierte Inhalte, darunter handschriftliche Merkmale wie Unterschriften oder Stempel, in ein standardisiertes Format überführen.

Ein weiteres Ziel ist der Vergleich unterschiedlicher Modellansätze hinsichtlich ihrer Extraktionsperformance und Effizienz. Hierbei wird ein kleineres, domänenspezifisch nachtrainiertes Modell gegen leistungsstärkere Modelle mit höherer Parameteranzahl antreten. Die Evaluation untersucht, ob ein Fine-Tuning mit einem begrenzten Datensatz vergleichbare oder bessere Ergebnisse erzielt als der Einsatz größerer Basismodelle.

Diese Arbeit konzentriert sich auf die Machbarkeit und die Evaluation der Modelle anhand eines Testdatensatzes. Die Entwicklung zielt auf einen funktionsfähigen Prototyp ab, der lokal betrieben wird. Eine vollständige Integration in das bestehende operative Fachverfahren ist nicht Gegenstand dieser Arbeit.

# Kapitel 2

## Theoretische Grundlagen

### 2.1 Dokumentarten

Im Rahmen der Kindergeldbeantragung für Auszubildende sind verschiedene Nachweise gültig. Zu den anerkannten Dokumententypen zählen der offizielle Vordruck der Bundesagentur für Arbeit (Formular KG5b)[Arbe 22] sowie Ausbildungsverträge. Die relevanten Informationen aus diesen Dokumenten sind von der fachlichen Seite vorgegeben. Zusätzlich laden Kunden häufig weitere Unterlagen, wie beispielsweise Schulbescheinigungen, im Portal hoch. Da diese für den Kindergeldantrag nicht im Fokus stehen, fallen sie im Folgenden in die Kategorie „Sonstiges“.

#### 2.1.1 KG5b-Formular

Das Formular KG5b ist, wie bereits erwähnt, ein officielles Dokument der Bundesagentur für Arbeit, welches als Bescheinigung der Ausbildungsstätte dient. Volljährige Kinder weisen damit gegenüber der Familienkasse den Status ihrer Ausbildung nach, was die Voraussetzung für den weiteren Kindergeldbezug ist.

Abbildung 2.1 zeigt ein exemplarisch ausgefülltes KG5b-Formular mit Markierung der für den Kindergeldantrag relevanten Felder.

#### 2.1.2 Ausbildungsvertrag

Im Gegensatz zu den standardisierten KG5b-Formularen weisen Ausbildungsverträge eine höhere Varianz auf. Dies ist auf die Vielzahl unterschiedlicher zuständiger Stellen (z. B. Industrie- und Handelskammern, Handwerkskammern, Ärztekammern) und Firmen zurückzuführen, die jeweils ein eigenes Layout definieren. Die Vielfalt der Dokumentenstruktur reicht dabei von formularbasierten Layouts bis hin zu unstrukturierten Fließtexten.

In der folgenden Abbildung 2.2 ist ein synthetischer Ausbildungsvertrag der Industrie- und Handelskammer[Hand 25] abgebildet, in dem die relevanten Felder markiert sind.

**(a) Erste Seite:**

- Angaben zum Kind:** Familiennname: Mustermann, Vorname: Max, Geburtsdatum: 01.03.2002.
- Angaben zum Ausbildungsverhältnis:**
  - apprenticeship\_finished:**  Das Ausbildungsverhältnis ist noch nicht beendet. (Bitte Nachweise beifügen)
  - start\_date\_apprenticeship:** Beginn der Ausbildung: 01.10.2025
  - end\_date\_apprenticeship:** Die Ausbildung dauert voraussichtlich bis 01.10.2026
  - exam\_month:** Monat der Abschlussprüfung (falls bekannt): September
- Bestätigung des Ausbildungsbetriebes:** stamp\_company (Apotheke am Markt, Marktstr. 1, 66666 Markt, Tel.: 66 666 6666 66), date\_document (28.07.2025), signature\_company (Unterschrift).

**(b) Zweite Seite:**

- Hinweis an den Kindergeldberechtigten:** Bitte füllen Sie Punkt e) erst aus, nachdem der Ausbildungsbetrieb die Angaben zum Ausbildungsverhältnis bestätigt hat!
- Punkt e):** Hat das Kind bei einem anderen Arbeitgeber ein Arbeitsverhältnis begonnen oder wird es demnächst ein Arbeitsverhältnis bei einem anderen Arbeitgeber beginnen?  ja  nein
- Wir versichern:** Ich versichere, dass unsere Angaben vollständig sind und der Wahrheit entsprechen. Uns ist bekannt, dass wir alle Änderungen, die für den Anspruch auf Kindergeld von Bedeutung sind, unverzüglich der Familienkasse mitteilen haben. Den Inhalt des Merkblattes Kindergeld (zu finden unter www.bzg.de oder www.familienkasse.de) haben wir zur Kenntnis genommen.
- Unterschriften:** Unterschrift der kindergeldberechtigten Person (Mustermann), Unterschrift des gesetzlichen Vertreters (Max Mustermann), Unterschrift des vollen Kindes (signature\_child).
- Aktionen:** Alle Eingaben löschen, Drucken, Speichern.

Abbildung 2.1: Exemplarisch ausgefülltes KG5b-Formular mit Markierung der relevanten Felder

### 2.1.3 Sonstige Dokumente

Die Kategorie **Sonstiges** steht als Auffangklasse für alle restlichen Dokumente bereit. Darin befinden sich zum Beispiel Schulbescheinigungen, Anträge auf Eintragung bei der Handelskammer oder Studienbescheinigungen. Da diese Dokumente keine Relevanz für die Weiterbeantragung des Kindergeldes bei volljährigen Auszubildenden haben, benötigt das System keine Informationen aus ihnen.

## 2.2 Fachliche Grundlagen

### 2.2.1 Der Ist-Zustand

Der schematische Ablauf der OCR/YOLO-Pipeline ist in Abbildung 2.3 dargestellt.

Der Prozess der Dokumentenverarbeitung lässt sich in zwei Stufen einteilen.

(a) Erste Seite

## (b) Zweite Seite

Abbildung 2.2: Synthetischer Ausbildungsvertrag der Industrie- und Handelskammer mit markierten Datenfeldern

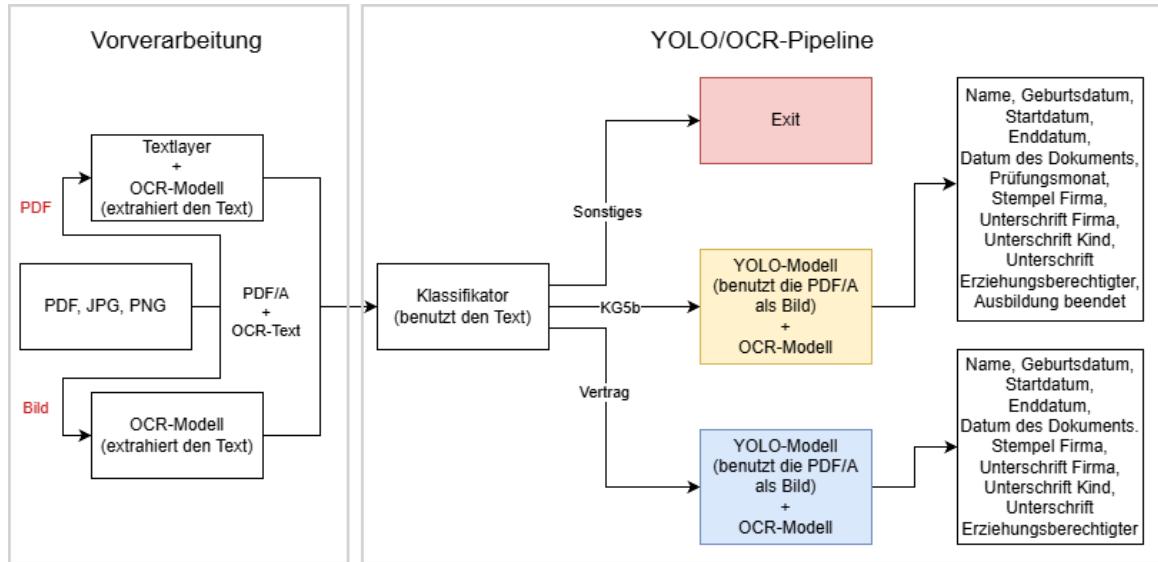


Abbildung 2.3: Schematischer Workflow der aktuellen OCR/YOLO-Pipeline zur Dokumentenverarbeitung

In der ersten Stufe nimmt die Anwendung die hochgeladenen Dokumente des Kunden entgegen. Wenn ein Dokument als PDF vorliegt, extrahiert die Pipeline den Text mithilfe des Textlayers und eines OCR-Modells, während bei einer Bilddatei lediglich das OCR-Modell zum Einsatz kommt. Die erste Stufe liefert ein PDF/A und den extrahierten Text als Ergebnis zurück.

Auf Basis des extrahierten Textes führt die Pipeline in der nächsten Stufe zunächst eine Klassifikation durch. Hierbei unterscheidet der Klassifikator zwischen den bereits vorgestellten Dokumententypen: KG5b, Vertrag und Sonstiges. Erkennt der Klassifikator das Dokument als **Sonstiges**, endet an dieser Stelle die Bearbeitung. Handelt es sich hingegen um ein KG5b-Formular oder einen Vertrag, startet das System je nach Dokumententyp eine Erkennung mit einem YOLO-Modell. Innerhalb der Bounding-Boxes extrahiert die Pipeline mit einem OCR-Modell den Text und ordnet ihn dem jeweiligen Label zu.

Schlussendlich stehen die erkannten Informationen zur weiteren Verarbeitung bereit.

## 2.2.2 Rahmenbedingungen und Infrastruktur

Die Entwicklung und Evaluation der Modelle erfolgt unter datenschutzrechtlichen Auflagen. Da die vorliegende Arbeit personenbezogene Echtdaten verarbeitet, kommt eine isolierte On-Premises-Infrastruktur zum Einsatz.

Die technische Architektur ist in Abbildung 2.4 schematisch dargestellt.

Das System basiert auf einem abgeschotteten Kubernetes-Cluster. Für die rechenintensiven Aufgaben, insbesondere das Fine-Tuning und die Inferenz der VLMs, stehen innerhalb des Clusters zwei NVIDIA A40 GPUs mit jeweils 48 GB VRAM [[NVID 22](#)] zur Verfügung. Um die Bereitstellung der VLMs zu erleichtern, nutzt das System Kubeflow. Kubeflow ist eine Open-Source-Plattform, die speziell für das Entwickeln, Trainieren und Deployen von Machine-Learning-Modellen konzipiert wurde.

Der Zugang zum Cluster sowie zur Integrated Development Environment (IDE) erfolgt über eine noVNC-Schnittstelle (browserbasierter Remote-Desktop).

Um einen Test- oder Trainingslauf durchzuführen, durchläuft das System den folgenden Prozess:

1. **Initialisierung:** Ein Python-Skript in der IDE startet den Kubeflow-Job.
2. **Datenbereitstellung:** Die Pipeline lädt sowohl das VLM als auch die zu klassifizierenden Dokumentenbilder aus dem S3-Store in den GPU-Cluster.
3. **Verarbeitung:** Die Inferenz oder das Training findet im Kubernetes-Cluster statt.

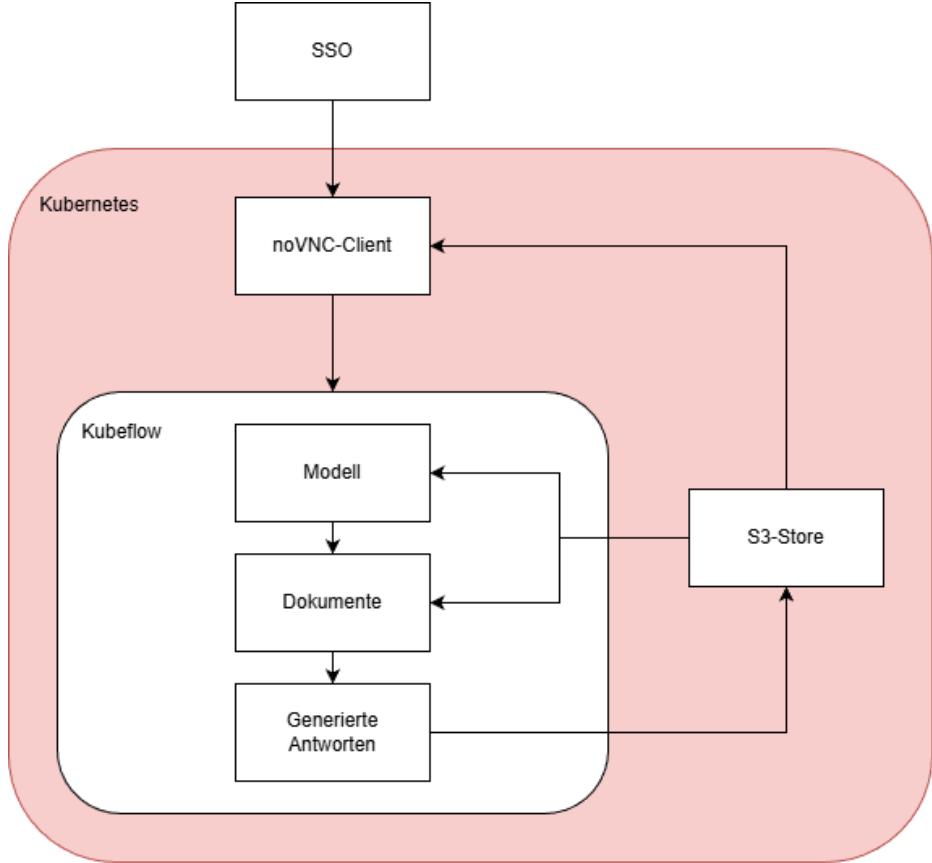


Abbildung 2.4: Schematische Darstellung der Trainings- und Inferenzinfrastruktur im Kubernetes-Cluster

4. **Persistierung:** Das System lädt die Ergebnisse zurück in den S3-Store.

## 2.3 Vision Language Models (VLMs)

Ein VLM besteht aus drei Komponenten: einem Image-Encoder, einem Adapter und einem Large Language Model (LLM). Abbildung 2.5 stellt den Aufbau dar.

Der Image-Encoder verarbeitet die Bildeingabe und extrahiert visuelle Merkmale (Features). Zu diesem Zweck kommen häufig vortrainierte Modelle wie Vision Transformer oder CLIP zum Einsatz. Diese Modelle zerlegen ein Bild in kleinere Bildausschnitte (Patches), die sie ähnlich wie Token in Sprachmodellen behandeln. Der Encoder überführt jeden Patch in einen Vektor, der die Eigenschaften repräsentiert. Das Ergebnis ist eine Sequenz von Bildvektoren, die die visuellen Informationen des gesamten Bildes beinhaltet.

Der Adapter verbindet den Image-Encoder mit dem Sprachmodell. Diese Schicht transformiert die Ausgabe des Image-Encoders in ein Format, das mit den Textvektoren des

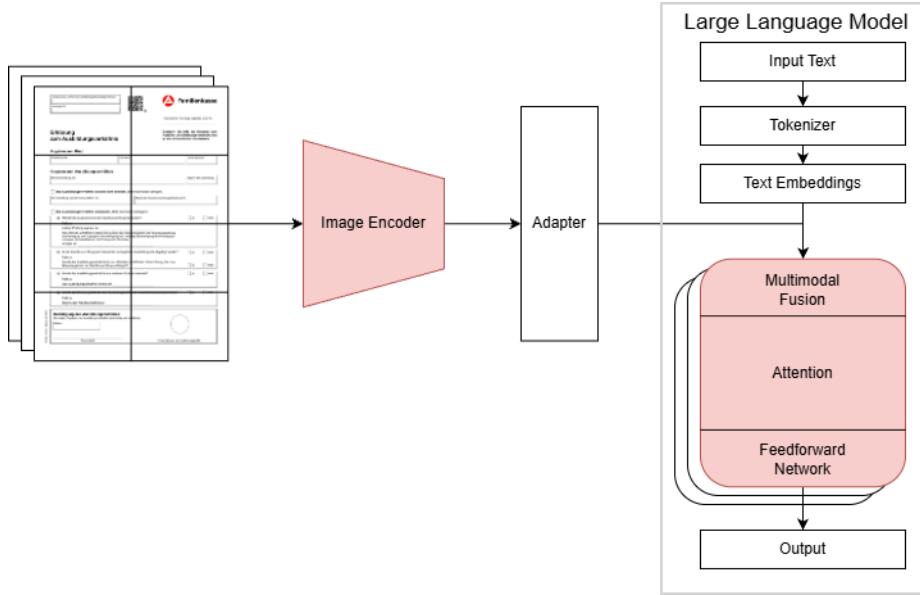


Abbildung 2.5: Architektur eines Vision Language Models mit Encoder, Adapter und LLM

Sprachmodells kompatibel ist. In vielen Modellen besteht der Adapter aus einer linearen Schicht oder einem kleinen neuronalen Netz. Der Adapter projiziert die Bildvektoren des Image-Encoders in denselben Vektorraum wie die Textvektoren des LLM.

Die Hauptkomponente bildet das LLM, welches für die eigentliche Verarbeitung und Generierung zuständig ist. Das LLM erhält die Textvektoren zusammen mit den Bildvektoren und fusioniert diese. Mithilfe des Attention-Mechanismus versteht das Modell Beziehungen zwischen visuellen und textuellen Elementen. So ist es möglich, Fragen zu Bildinhalten zu beantworten oder Bildbeschreibungen zu generieren.

Die größte Herausforderung dieser modernen Technologie ist der hohe VRAM-Verbrauch. Neben der statischen VRAM-Nutzung kommt mit jeder Anfrage ein dynamischer Verbrauch dazu. Dabei setzt sich der statische Verbrauch aus den Gewichten der Modelle zusammen, während sich der dynamische Verbrauch aus den Key-Value-Caches und den Aktivierungen bildet.

## 2.4 Vorstellung der Modelle

Für die Klassifikation und die Extraktion der Informationen aus den Dokumenten evaluiert diese Arbeit drei verschiedene VLMs.

### 2.4.1 Pixtral-12B

Mistral AI veröffentlichte das Pixtral-12B-2409[Pixt 24] im Jahr 2024. Es basiert auf einem 12 Milliarden Parameter großen Text-Decoder mit einem zusätzlichen 400 Millionen Parameter umfassenden Vision-Encoder. Das Modell wurde speziell auf das Verständnis von Bildern und Dokumenten trainiert, weshalb es einen optimalen Kandidaten für die vorliegende Arbeit darstellt. Mit einem theoretischen Kontextfenster von 128.000 Token ermöglicht das Modell die gleichzeitige Verarbeitung mehrerer Bilder[Agra 24].

### 2.4.2 Qwen-2.5-VL

Alibaba Cloud veröffentlichte das Qwen2.5-VL-7B-Instruct[Qwen 25b] sowie das Qwen2.5-VL-32B-Instruct[Qwen 25a] im Jahr 2025. Die beiden Modelle basieren auf dem gleichen Image-Encoder mit 600 Millionen Parametern. Lediglich die Größe des Text-Decoders ist mit 7 Milliarden beziehungsweise 32 Milliarden Parametern unterschiedlich. Eine Besonderheit dieser Modelle ist, dass sie speziell auf das Verarbeiten von Dokumenten trainiert wurden[Bai 24].

### 2.4.3 Benchmarks

Im Folgenden sind die Benchmarks der einzelnen Modelle gelistet. Besonders das Ergebnis des DocVQA-Benchmarks ist von Interesse, da der Benchmark ein Visual Question Answering (VQA) auf Dokumentenbildern durchführt[Mine 21].

Modell	Pixtral-12B	Qwen2.5-VL-7B	Qwen2.5-VL-32B
DocVQA	90,7	95,7	94,8
MMMU	52,0	58,6	70,0

Tabelle 2.1: Benchmark-Ergebnisse der evaluierten Modelle[Bai 24, Qwen 25a, Agra 24]

## 2.5 Parameter-Efficient Fine-Tuning

Das Anpassen der Gewichte eines bereits trainierten Modells an eine spezifische Domäne setzt eine Infrastruktur mit hoher Rechenleistung voraus. Mithilfe von Parameter-Efficient Fine-Tuning (PEFT) lässt sich die Anzahl der zu trainierenden oder neuen Parameter senken. Infolgedessen verringert sich auch der Rechenaufwand für das Fine-Tuning der Modelle[Mang 22].

Eine der bekanntesten Methoden im Bereich PEFT ist die Low-Rank Adaptation (LoRA). Anstatt das gesamte Modell neu zu trainieren, friert LoRA die ursprünglichen Gewichte ein

und fügt stattdessen trainierbare Matrizen mit einem niedrigen Rang  $r$  in jede Schicht der Transformer-Architektur ein. Um Rechenaufwand und Speicherplatz zu sparen, trainiert der Algorithmus ausschließlich diese kleineren Matrizen [Hu21].

Ergänzend zu LoRA betrachtet diese Arbeit die Variante Rank-Stabilized LoRA (rsLoRA). Bei der Standard-LoRA führt ein steigender Rang  $r$  oft nicht zu einer besseren Performance, da der verwendete Faktor  $(\alpha/r)$  das Lernen bei höheren Rängen verlangsamen oder hemmen kann. rsLoRA stabilisiert den Prozess, indem das Verfahren die Adapter durch die Quadratwurzel des Rangs  $(\alpha/\sqrt{r})$  teilt [Kala23].

## 2.6 Verwandte Arbeiten

Das automatisierte Lesen, Verstehen und Analysieren von Dokumenten entwickelt sich zu einem zunehmend relevanten Forschungsfeld, das die Literatur häufig unter dem Begriff Document AI zusammenfasst. Cui et al. liefern eine Übersicht über die Entwicklung dieses Bereichs und definieren den Begriff Document AI als Prozess, der Webseiten, digitale und gescannte Dokumente in strukturierte Information überführt [Cui21]. Die Entwicklung lässt sich dabei in drei wesentliche Phasen einteilen, die die folgenden Unterkapitel besprechen.

### 2.6.1 Die Grenzen sequentieller OCR-Pipelines

Frühe Ansätze im Bereich der Document AI beruhten auf Pipelines, bei denen ein OCR-Modell den Text extrahierte. Ein Problem besteht darin, dass herkömmliche OCR-Modelle den Text als zusammenhängende Zeichenkette verarbeiten, wodurch die zweidimensionalen Layout-Informationen, wie die räumliche Anordnung von Tabellen und Textblöcken, verloren gehen.

Um diese Informationen zu behalten, stellten Katti et al. mit Chargrid einen Ansatz vor, der Dokumente als zweidimensionales Raster von Zeichen (Character Grid) repräsentiert. Durch den Einsatz von Convolutional Neural Networks auf dieser Struktur konnte das Modell sowohl textuelle als auch räumliche Merkmale nutzen, was zu einer Leistungssteigerung bei Aufgaben wie der Informationsextraktion (Information Extraction) aus Rechnungen führte [Katt18]. Chargrid baut auf den Bounding-Boxen und Koordinaten der OCR-Modelle auf und benutzt diese, um das Layout des Dokuments zu modellieren.

Dieser Ansatz der Verarbeitung des Textes mit einem OCR-Modell ähnelt dem Klassifikationsschritt der aktuellen OCR/YOLO-Pipeline, die das Dokument ebenfalls ohne räumliche Merkmale klassifiziert. Die Einbeziehung räumlicher Merkmale ist für die Robustheit notwendig, was die Fehleranfälligkeit insbesondere bei komplexen Layouts erklärt.

### 2.6.2 Semantische Layout-Modellierung

Mit der Vorstellung der Transformer-Architektur begann eine tiefere Analyse der Semantik. Xu et al. erkannten, dass bisherige Lösungen sich fast ausschließlich auf Text konzentrieren, während das Layout eines Dokuments eine entscheidende Rolle für das Verständnis spielt. LayoutLM verbindet ein vortrainiertes BERT-Modell mit Positionsvektoren und Bildvektoren. Die Positionsvektoren speichern die räumliche Anordnung der Elemente, während die Bildvektoren visuelle Merkmale des Dokuments kodieren. LayoutLM konnte auf diversen Benchmarks neue Bestwerte erzielen [Xu 20].

Einen weiteren Schritt markierte das TrOCR von Li et al., bei dem Transformer sowohl die CNN-basierten Komponenten für das Bildverständnis als auch die Recurrent Neural Networks (RNNs) für die Textgenerierung herkömmlicher OCR-Modelle ersetzten. Dies führte ebenfalls zu neuen Bestwerten in der Texterkennung [Li 23].

Trotz dieser Fortschritte bleibt bei LayoutLM und ähnlichen Modellen die Trennung zwischen Textextraktion durch ein OCR-Modell und anschließender semantischer Verarbeitung bestehen. Mit dieser Trennung stützt sich die Verarbeitung weiterhin auf mehrere Modelle, was die Abhängigkeit und Fehleranfälligkeit erhöht. Jeder OCR-Fehler beeinträchtigt die anschließende semantische Analyse. Diese Abhängigkeit unterschiedlicher Modelle ist auch eine Schwachstelle der OCR/YOLO-Pipeline, weshalb ein End-To-End-Ansatz Abhilfe schaffen könnte.

### 2.6.3 End-To-End-Architekturen und multimodale Ansätze

Um die Abhängigkeit und Fehleranfälligkeit sequentieller Pipelines zu reduzieren, entwickelte sich die Forschung hin zu End-To-End-Architekturen. Kim et al. stellten mit Donut ein OCR-freies Transformer-Modell vor, das durch einen Vision-Encoder und einen Textual-Decoder Bilder von Dokumenten direkt in eine strukturierte JSON-Ausgabe überführt. Durch den Wegfall des OCR-Modells erreicht Donut eine hohe Genauigkeit bei gleichzeitig geringerer Latenz [Kim 22]. Die Robustheit eines solchen OCR-freien Systems zeigten Blecher et al. anhand akademischer Dokumente mit Nougat. Durch die Umwandlung von Dokumenten in Markdown ohne die Verwendung eines OCR-Modells extrahiert Nougat mathematische Formeln ohne Verlust. Traditionelle OCR-Systeme scheiterten häufig an dieser Aufgabe, da besonders hier räumliche Zusammenhänge relevant sind [Blec 23].

Parallel dazu trieb die Forschung die Integration von Dokumentenverarbeitung in LLMs voran. Wang et al. präsentierten mit DocLLM eine Erweiterung für LLMs, die auf teure Image-Encoder verzichtet. DocLLM kombiniert die aus einem OCR-Modell gewonnenen Bounding-Box-Koordinaten über einen Attention-Mechanismus mit dem Modell, wodurch es die Struktur des Dokuments versteht [Wang 23].

Die Entwicklung von Donut und Nougat zeigt das Potenzial OCR-freier Systeme. DocLLM demonstriert, wie sich das Wissen von LLMs für Aufgaben im Bereich der Document AI nutzbar machen lässt. Die aktuellen VLMs wie das Pixtral-12B und die Qwen-2.5-VL-Familie führen diese beiden Systeme zusammen. Sie vereinen die Image-Encoder mit dem Wissen des LLMs und können visuelle und textuelle Merkmale verbinden. Ähnlich wie die Evolution der Document AI beschäftigt sich diese Arbeit damit, ob ein solches generalistisches VLM die auf OCR-Modellen beruhende Pipeline ersetzen kann.

# Kapitel 3

## Methodik und Implementierung

Um die Eignung der VLMs als potenziellen Ersatz für die OCR/YOLO-Pipeline zu beurteilen, werden die folgenden Forschungsfragen beantwortet:

1. Wie verhält sich ein nicht domänen spezifisch angepasstes VLM hinsichtlich Latenz, Klassifikations- und Extraktionsleistung im Vergleich zur OCR/YOLO-Pipeline, und welche Vorteile bietet ein einzelnes generalistisches Modell gegenüber spezialisierten Einzelsystemen?
2. Welche Auswirkungen hat ein domänen spezifisches Training eines kleineren Modells auf dessen Latenz, Klassifikations- und Extraktionsleistung im Vergleich zu einem leistungs stärkeren Basismodell?
3. Wie unterscheidet sich der Ressourcenverbrauch (Energieverbrauch und VRAM-Auslastung) der VLMs von dem der OCR/YOLO-Pipeline?

Abbildung 3.1 zeigt einen Überblick der gesamten Methodik.

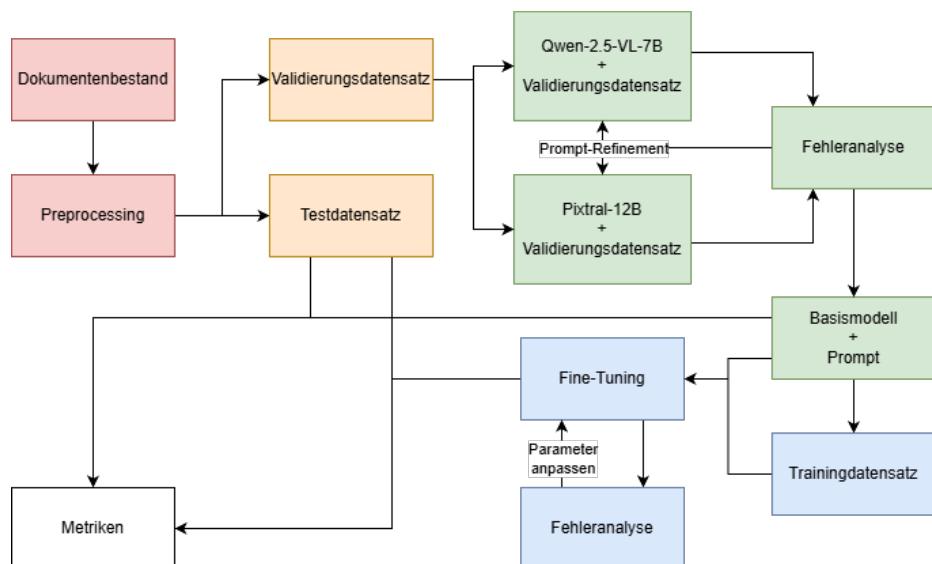


Abbildung 3.1: Überblick der Methodik

## 3.1 Modellierung des Dokumentenbestands als Grundlage der Evaluation

### 3.1.1 Preprocessing des Dokumentenbestands

Durch Datenlieferungen der Familienkasse stehen für die Evaluation mehrere Tausend Dokumente als PDF oder Bilddatei zur Verfügung. Um einen einheitlichen Dokumentenkorpus zu schaffen und die Klassifikations- und Extraktionsleistung zu verbessern, erfolgt eine Vorverarbeitung der Dokumente. Dieser Prozess ist in Abbildung 3.1 in Rot dargestellt.

Im Fokus des Preprocessings stehen die Korrektur der Bildausrichtung sowie die Konvertierung der PDF-Dokumente in Bilddateien. Zur Korrektur der Rotation werden die Exchangeable Image File Format (EXIF)-Metadaten ausgelesen, um die ursprüngliche Ausrichtung wiederherzustellen. Während ältere Modelle Bilder auf eine feste Dimension skalieren, unterstützen die in dieser Arbeit evaluierten Modelle eine dynamische Skalierung. Zur Begrenzung der maximalen Token-Anzahl werden die Bilder dennoch auf eine maximale Pixelanzahl skaliert, wobei das ursprüngliche Seitenverhältnis beibehalten wird. Listing 1 zeigt die Korrektur der Bildausrichtung und Skalierung innerhalb des Preprocessings.

---

```

from pathlib import Path
from PIL import Image, ImageOps
from pdf2image import convert_from_path

def preprocessing(path_to_document: str, max_pixels: int) -> [Image]:
    ...
    for _, page in enumerate(pages):
        # corrects the image orientation
        page = ImageOps.exif_transpose(page)

        if page.width * page.height > max_pixels:
            scale_factor = (max_pixels / (page.width * page.height)) ** 0.5

            new_width = int(page.width * scale_factor)
            new_height = int(page.height * scale_factor)

            page_resized = page.resize((new_width, new_height), Image.Resampling.LANCZOS)
    ...

```

---

Listing 1: Preprocessing der Dokumente

### 3.1.2 Modellierung der Dokumentenarten als JSON-Objekte

Für die standardisierte Weiterverarbeitung der Modellausgaben ist eine konsistente Struktur essenziell. Da moderne VLMs darauf trainiert sind, Antworten im JavaScript Object Notation (JSON)-Format zu liefern, werden die extrahierten Informationen in ein vordefiniertes JSON-Schema überführt.

Ein wesentlicher Vorteil gegenüber der herkömmlichen Pipeline besteht in der simultanen Klassifikation sowie der Information Extraction (IE) in einem einzigen Inferenzschritt. Die JSON-Schemata für die relevanten Dokumentenarten sind in den Listings 2 und 3 dargestellt. Bei dem JSON-Schema der sonstigen Dokumente in Listing 4 findet nur die Klassifikation statt, da hier keine Informationen relevant sind. Die Felder der JSON-Schemata geben die relevanten Informationen aus Abschnitt 2.1 wieder.

---

```
{
  "file_name": {
    "type": "kg5b",
    "name_child": "name, vorname",
    "birthday_child": "DD.MM.YYYY",
    "start_date_apprenticeship": "DD.MM.YYYY",
    "end_date_apprenticeship": "DD.MM.YYYY",
    "date_document": "DD.MM.YYYY",
    "stamp_company": true,
    "signature_company": true,
    "signature_child": true,
    "signature_legal_guardian": true,
    "apprenticeship_finished": true,
    "exam_month": "MM"
  }
}
```

---

Listing 2: JSON-Schema des Dokumententyps KG5b

---

```
{
  "file_name": {
    "type": "vertrag",
    "name_child": "name, vorname",
    "birthday_child": "DD.MM.YYYY",
    "start_date_apprenticeship": "DD.MM.YYYY",
    "end_date_apprenticeship": "DD.MM.YYYY",
    "date_document": "DD.MM.YYYY",
    "stamp_company": true,
    "signature_company": true,
    "signature_child": true,
    "signature_legal_guardian": true
  }
}
```

---

Listing 3: JSON-Schema des Dokumententyps Ausbildungsvertrag

---

```
{
  "file_name": {
    "type": "sonstiges"
  }
}
```

---

Listing 4: JSON-Schema des Dokumententyps Sonstiges

Die Klassifikation wird im JSON-Objekt über das Feld `type` abgebildet und ist in allen JSON-Schemata vorhanden. Felder der Information Extraction sind über die Dokumentenarten nicht konsistent, jedoch bildet das Schema eines Vertrags eine Teilmenge des Schemas eines KG5b-Formulars. Das Schema des KG5b-Formulars beinhaltet zusätzlich die Felder `apprenticeship_finished` und `exam_month`. `apprenticeship_finished` ist ein binäres Feld und bildet die Checkboxen auf der ersten Seite des Formulars ab (siehe Abbildung 2.1a).

Bei den binären, booleschen Feldern markiert ein `true` das Vorhandensein eines Merkmals, während ein `false` dessen Fehlen oder das Nicht-Erkennen repräsentiert. Um eine strukturelle Konsistenz zu erzwingen, werden Felder, bei denen die dazugehörige Information im Dokument fehlt, bei booleschen Typen mit `false` und bei Textfeldern mit einer leeren Zeichenkette belegt.

Da die Dokumente in den meisten Fällen aus mehreren Seiten bestehen, repräsentiert ein JSON-Objekt das mehrseitige Dokument.

### 3.1.3 Test- und Validierungsdatensatz

Die Erstellung des Test- sowie des Validierungsdatensatzes ist im methodischen Überblick (Abbildung 3.1) orange hervorgehoben und schließt unmittelbar an die Vorverarbeitung der Dokumente an. Beide Datensätze umfassen jeweils 60 Dokumente, aufgeteilt in 20 Beispiele pro Dokumentenart. Um die reale Datenverteilung bestmöglich abzubilden, werden die Dokumente manuell ausgewählt.

Die Datensätze decken dabei verschiedene Herausforderungen ab. Neben Dokumenten mit geringer Qualität, vorrangig mangelhaften Scans und Fotos, befinden sich fehlerhaft aus gefüllte Dokumente sowie Dokumente mit fehlenden Unterschriften und Stempeln in den Datensätzen. Um die hohe Varianz der Verträge und sonstigen Dokumente abzubilden, enthalten die Datensätze viele unterschiedliche Layouts. Die Exemplare für die Verträge beinhalten sowohl starre Layouts als auch Fließtexte unterschiedlicher Firmen und Kammern. Die sonstigen Dokumente beinhalten keine kontextfremden Dokumente, sondern Schulbescheinigungen, Studienbescheinigungen und Eintragungen bei der Handwerkskammer, um die Komplexität der Unterscheidung zu erhöhen.

Zur Erstellung der Wahrheitswerte (Ground Truth) werden alle Dokumente manuell annotiert und in das entsprechende JSON-Schema überführt.

## 3.2 Evaluation der generierten JSON-Objekte

### 3.2.1 Validierung des VLM-Outputs

Um eine korrekte Formatierung des JSON-Outputs der VLMs sicherzustellen, existieren verschiedene Ansätze. Während komplexere Lösungen nur Tokens zulassen, die syntaktisch in JSON-Objekten enthalten sein könnten (Constrained Decoding), wird in dieser Evaluation ein direkterer Ansatz gewählt. Hierbei werden aus der generierten Antwort potenzielle JSON-Objekte extrahiert und geparsst. Schlägt das Parsen der JSON-Objekte fehl, werden diese zurück an das VLM geliefert mit der Anweisung, diese zu korrigieren. Das Modell bekommt insgesamt drei Versuche, ein valides JSON-Objekt zu generieren, bevor das Dokument als ungültig eingestuft wird.

### 3.2.2 Vergleichslogik der unterschiedlichen JSON-Felder

Im Gegensatz zur Verarbeitung von Antworten herköModels weisen VLMs eine hohe Varianz an Ausgabeformaten auf. Verschiedene Feldtypen der generierten JSON-Objekte müssen deshalb spezifisch normalisiert werden, um eine Grundlage für den Vergleich zu bilden. Des Weiteren müssen Feldtypen wie Namen, Booleans, Zeichenketten, Datumsangaben und

Monate nach unterschiedlichen Logiken verglichen werden. Für den Vergleich werden auf Basis der Klasse in Listing 5 verschiedene Komparatoren (Comparator) definiert.

---

```
from typing import Dict

class Comparator:
    def compare(self, pred_value, gt_value) -> Dict:
        pass
```

---

Listing 5: Basisklasse der verschiedenen Comparator

Der NameComparator in Listing 6 vergleicht die Namensfelder mithilfe der Levenshtein-Similarity, um kleinere Fehler und verschiedene Schreibweisen zu tolerieren. Beispielsweise wird ein Name, der ue statt ü enthält, nicht als Fehler gewertet. Der Threshold, der zwischen korrekt und nicht korrekt entscheidet, liegt bei 0,8. Da die OCR/YOLO-Pipeline denselben Vergleich mit identischem Threshold verwendet, wird eine Vergleichbarkeit der Ergebnisse gewährleistet.

---

```
from rapidfuzz import fuzz

class NameComparator(Comparator):
    def __init__(self, threshold=0.80):
        self.threshold = threshold

    def compare(self, pred_value: str, gt_value: str) -> Dict:
        # calculate levenshtein-similarity
        similarity = fuzz.ratio(pred_value.lower(), gt_value.lower()) / 100.0
        is_correct = similarity >= self.threshold

    ...
```

---

Listing 6: Comparator für Namen

Der BooleanComparator normalisiert boolesche Felder entsprechend der Python-Syntax zu True und False. Zu den akzeptierten Varianten gehören sowohl true und false, 0 und 1, wahr und falsch als auch vorhanden und nicht vorhanden.

Zeichenketten, die keine Namen darstellen, werden durch den ExactComparator verglichen. Hierbei findet keine Normalisierung statt, sodass die Zeichenketten exakt übereinstimmen müssen.

Datumsangaben werden durch den `DateComparator` einheitlich in das Format DD.MM.YYYY überführt. Obwohl dies nicht dem internationalen Standard entspricht, erweist sich dieses Vorgehen als am robustesten, da die vorliegenden deutschen Dokumente primär in diesem Format ausgefüllt sind.

Der `MonthComparator` normalisiert alphanumerische Monate in eine numerische Darstellung. Falls vollständige Datumsangaben geliefert werden, wird der `DateComparator` genutzt und anschließend der Monat extrahiert.

Um die verschiedenen Komparatoren den Feldern zuzuordnen, existiert für jede Dokumentenart eine Konfiguration, die dem jeweiligen JSON-Schema aus Abschnitt 3.1.2 entspricht. Das Schema 7 definiert beispielhaft die Konfiguration für das KG5b-Formular.

---

```
import ExactComparator, NameComparator, DateComparator, BooleanComparator, MonthComparator

CONFIG_KG5B = {
    "type": {"type": "string", "comparator": ExactComparator()},
    "name_child": {"type": "string", "comparator": NameComparator()},
    "birthday_child": {"type": "date", "comparator": DateComparator()},
    "start_date_apprenticeship": {"type": "date", "comparator": DateComparator()},
    "end_date_apprenticeship": {"type": "date", "comparator": DateComparator()},
    "date_document": {"type": "date", "comparator": DateComparator()},
    "stamp_company": {"type": "boolean", "comparator": BooleanComparator()},
    "signature_company": {"type": "boolean", "comparator": BooleanComparator()},
    "signature_child": {"type": "boolean", "comparator": BooleanComparator()},
    "signature_legal_guardian": {"type": "boolean", "comparator": BooleanComparator()},
    "apprenticeship_finished": {"type": "boolean", "comparator": BooleanComparator()},
    "exam_month": {"type": "date", "comparator": MonthComparator()}
}
```

---

Listing 7: Konfiguration des KG5b-Formulars

### 3.2.3 Bewertung der Klassifikation und der Information Extraction

Um die generierten JSON-Objekte der VLMs schlussendlich bewerten zu können, werden die Klassifikation und Information Extraction unabhängig voneinander bewertet. Für die Evaluation werden die einzelnen JSON-Felder jedes Dokuments mithilfe der in der Konfiguration definierten Komparatoren mit der Ground Truth verglichen und verschiedene Metadaten erfasst. Die Metadaten beinhalten den Status (vorhanden, nicht vorhanden oder halluziniert), die Korrektheit sowie die Fehlerart des Feldes. Zu den Fehlerarten gehören Wert- und Formatfehler. Halluzinierte Felder sind definiert als Felder, die nicht in der jeweiligen Konfiguration beziehungsweise im Schema vorkommen. Das Ergebnis ist ein Python-

Dictionary (Listing 8), das für jedes JSON-Objekt eines Dokuments die Metadaten für jedes Feld beinhaltet.

---

```

result = {
    "document1": {
        "type": {
            "status": "present",
            "is_correct": True,
            "error": None
        },
        "name_child": {
            "status": "missing",
            "is_correct": None,
            "error": None
        },
        "birthday_legal_guardian": {
            "status": "hallucinated",
            "is_correct": None,
            "error": None
        }
    },
    ...
}

```

---

Listing 8: Ergebnis-Dictionary mit den Metadaten des gesamten Dokumentenkorpus

Auf Basis dieses Dictionaries wird für die Bewertung der Klassifikation aus jedem Dokument das Feld `type` extrahiert. Da das Feld `type` in jeder JSON der Dokumentenarten (siehe Listings 2, 3 und 4) vorkommt, kann es nicht halluziniert werden. Jedoch kann das Feld fehlen, weshalb neben den Klassen KG5b, Vertrag und Sonstiges eine weitere Klasse `missing` eingeführt wird. `Missing` wird in der Auswertung nur betrachtet, wenn mindestens ein Dokument in der Klasse vorkommt. Für die Bestimmung der Güte der Modelle wird eine Confusion-Matrix erstellt, wobei der Fokus auf dem F1-Score liegt.

Die Bewertung der Information Extraction ist komplexer als die der Klassifikation. Um den Entity-Level F1-Score, der die Hauptmetrik für den Vergleich darstellt, zu berechnen, werden die Metadaten aller Felder außer dem Feld `type` benötigt.

Auf Basis dieser Metadaten werden für den gesamten Dokumentenkorpus die True Positives (TP), False Positives (FP) und False Negatives (FN) gezählt. Ein Feld zählt als TP, wenn es vorhanden sowie korrekt ist. Zu den FN gehören Felder, die entweder fehlen oder vorhanden, aber falsch sind. Ein Feld, das vorhanden, aber falsch ist, zählt zusätzlich zu den FP, zusammen mit den Feldern, die halluziniert werden. Diese doppelte Bestrafung stellt eine

sehr strenge Bewertung für inhaltliche Fehler dar. True Negatives (TN) sind nichtzählbar, da die Menge an nicht gefundenen, leeren Feldern theoretisch unendlich groß ist. Durch die Schemata und die Ground Truths ist eindeutig feststellbar, welche Felder halluziniert sind. Da mithilfe der Schemata auch ohne Ground Truth die benötigten Felder gefiltert werden können, kann wie in Listing 9 entschieden werden, ob die Halluzinationen generell bestraft werden.

---

```
def calculate_field_based_f1_score(gt_jsons,
                                    pred_jsons,
                                    penalize_halluzinations: bool = False) -> float:
    ...
    fp = count_correct_fields

    if penalize_halluzinations:
        fp += count_halluzinated_fields
    ...

```

---

Listing 9: Parameter, der bestimmt, ob Halluzinationen bestraft werden

### 3.3 Messung von Latenz, Energieverbrauch und VRAM-Auslastung

Im Hinblick auf die Bewertung der Modelle werden neben der Güte der Klassifikation und Information Extraction die Latenz, die VRAM-Auslastung sowie der Energieverbrauch gemessen.

Die Latenz der Modelle stellt lediglich das Delta zwischen Start- und Endzeit der Inferenz dar. Hierbei werden die Zeitpunkte gemessen, an denen die eigentliche Inferenz startet und die Antwort bereitsteht. Das Laden des Modells wird nicht betrachtet, da dieses keinen Einfluss auf die Antwortzeit in der produktiven Umgebung hat und folglich nicht relevant für die Evaluation ist.

Des Weiteren wird der Energieverbrauch während der Inferenz in Joule erfasst. Die Messung der Leistung in Watt ist nicht zielführend, da sie die Zeit nicht berücksichtigt und Modelle mit geringerer Latenz bei vergleichbarem Energieverbrauch benachteiligt werden.

Ebenso wie der Energieverbrauch wird der dynamische VRAM-Verbrauch während der Inferenz gemessen. Mit der Initialisierung des Modells wird zudem einmalig der statische Verbrauch erfasst.

### 3.4 Auswahl des Basismodells

Die Auswahl eines geeigneten Basismodells ist ein wichtiger Schritt für das spätere Fine-Tuning. Wie in Abbildung 3.1 (grün markierte Felder) dargestellt, erfolgt diese Auswahl in einem iterativen Prozess. Ziel ist es, das Modell zu identifizieren, das bereits ohne Training die beste Leistung sowohl bei der Klassifikation als auch bei der Information Extraction erreichen kann.

In jeder Iteration werden die Modelle Pixtral-12B und Qwen-2.5-VL-7B mit dem Validierungsdatensatz 3.1.3 evaluiert. Die Temperatur der Modelle wird auf null gesetzt, um Halluzinationen zu vermeiden. Jede Iteration gliedert sich in folgende Schritte:

1. Evaluation: Durchführung der Inferenz beider Modelle mit dem Validierungsdatensatz.
2. Fehleranalyse: Untersuchung der Fehler in der Klassifikation und in der Information Extraction.
3. Prompt-Refinement: Anpassung des Prompts auf Basis der Fehleranalyse.

Der Prozess wird so lange durchlaufen, bis durch Änderungen des Prompts keine Steigerungen der Metriken mehr erzielt werden. Im Anschluss werden mit dem finalen Prompt und dem Testdatensatz 3.1.3 die finalen Metriken bestimmt und auf deren Basis das Modell gewählt. Die Trennung von Validierungs- und Trainingsdatensatz schließt ein Data Leakage aus und verhindert ein Overfitting auf den Testdaten.

Bezogen auf das Prompt-Engineering wird bewusst ein Zero-Shot-Ansatz gewählt. Im Gegensatz zu One-Shot- und Few-Shot-Prompting werden hier weniger Tokens verbraucht, was die Kosten pro Inferenz senkt. Des Weiteren kann das Modell so schneller antworten, da es weniger Input-Tokens verarbeiten muss.

Die Gestaltung des finalen Prompts (siehe Listing 10) folgt der R-K-F-Formel. Während der Kontext die vorliegenden Dokumentarten beschreibt und die Schemata definiert, gibt das Format die Ausgabeformatierung vor. Zum Einsparen von Tokens wird das Modell strikt angewiesen, lediglich das JSON-Objekt zu generieren.

Der hier evaluierte Prompt wird im Laufe der Arbeit auch für die Bewertung des Qwen-2.5-VL-32B genutzt.

---

```

prompt = """
    Du bist ein Dokumenten-Assistent.

    Analysiere die Dokumente und extrahiere die Daten im JSON-Format.
    Gib ausschließlich valides JSON zurück.
    Es werden folgende Dokumentarten unterschieden:

        1. KG5b: Ein offizielles Dokument der Bundesagentur für Arbeit zur Bescheinigung des
               Ausbildungsstatus. Wichtig: Der 'exam_month' entspricht nicht zwangsläufig dem
               'end_date_apprenticeship'. Das 'date_document' ist das Datum, an dem der
               Ausbildungsbetrieb unterschrieben hat.
        2. Vertrag: Ein klassischer Ausbildungsvertrag.
        3. Sonstiges: Alle Dokumente, die weder ein KG5b noch ein Vertrag sind.

    Schemas für die Dokumentarten:

    ...

    Extrahiere die Informationen aus den Bildern und wähle das passende Schema für die
    jeweilige Dokumentenart aus.
    Sollten mehrere Dokumentenarten vorliegen, erstelle für jede Art ein separates
    JSON-Objekt mit dem entsprechenden Schema.
    Fülle ausschließlich die im Schema definierten Felder aus.
    Gib keine weiteren Texte aus, sondern nur das valide JSON.

"""

```

---

Listing 10: Finaler Prompt der Modellauswahl

## 3.5 Fine-Tuning des Basismodells

### 3.5.1 Trainingsdatensatz

Um ein qualitativ hochwertiges Fine-Tuning zu ermöglichen, ist ein umfangreicherer Datensatz als für die Modellauswahl erforderlich. Der finale Trainingsdatensatz umfasst 610 Dokumente, die sich aus 227 Verträgen, 165 KG5b-Formularen und 218 sonstigen Dokumenten zusammensetzen.

Die Klassen sind bewusst ungleich verteilt. KG5b-Formulare haben ein starres Layout, wodurch schon mit einer geringeren Anzahl an Trainingsdaten ein gutes Ergebnis erwartet wird. Infolge der hohen Varianz der Verträge und der sonstigen Dokumente erhalten diese im Training eine höhere Gewichtung. Anders als beim Testdatensatz werden hier die Exemplare nicht manuell, sondern mithilfe einer Stichprobe ausgewählt. Dabei werden die Daten aus einem einwöchigen Zeitraum betrachtet, wodurch ohne manuelle Auswahl eine

genaue Repräsentation der realen Daten entsteht. Des Weiteren wird so ein Selection Bias verhindert.

Zur effizienten Erstellung der Ground Truth wird ein Model-Assisted Labeling eingesetzt. Hierbei generiert das im Prozess der Modellauswahl 3.4 gewonnene Modell, zusammen mit dem angepassten Prompt, für jedes Dokument das jeweilige JSON-Objekt. Im Anschluss werden diese JSON-Objekte manuell kontrolliert und korrigiert. Durch den Einsatz des Modells zur Ermittlung der Ground Truth kann der Aufwand gesenkt werden.

Für das Fine-Tuning muss der Datensatz in das OpenAI Message Format (Listing 11) gebracht werden. Des Weiteren müssen die JSON-Objekte in Markdown-Blöcke eingebettet werden, um dem bereits gelernten Format des VLM zu entsprechen.

---

```
import json

response = f"""json.dumps(ground_truth)"""

messages = [{"messages": [{"role": "user",
    "content": [{"type": "text", "text": prompt},
                {"type": "image", "image": image}],
    ...
  ],
  ...
  },
  {"role": "assistant",
    "content": [{"type": "text", "text": response}]
}
]

},
...
]
```

---

Listing 11: OpenAI Message Format

### 3.5.2 Optimierung des Fine-Tunings

Das in Abschnitt 3.4 gewählte Basismodell wird mittels PEFT an die Dokumentenarten angepasst. Dabei kommen LoRA und die stabilisierte Variante rsLoRA zum Einsatz. Für das Training wird auf das Framework Unislot gesetzt, um den Ressourcenverbrauch gering zu halten. Des Weiteren wird das Modell in 4-Bit-Quantisierung geladen, um die VRAM-Auslastung weiter zu senken.

Der Systemprompt in Listing 12 ist pragmatisch gehalten, damit das Modell anhand von Beispielen statt durch Instruktion lernt.

---

```

prompt = """
    Extrahiere die relevanten Informationen aus den Dokumenten und gebe diese als JSON zurück.
"""

```

---

Listing 12: Prompt für das Fine-Tuning des Qwen-2.5-VL-7B

Der Trainingsdatensatz 3.5.1 wird mithilfe von Datenaugmentation an das Training angepasst. Vor jedem Trainingslauf wird der Datensatz vollständig randomisiert, um eine zufällige Reihenfolge der Trainingsdaten zu erhalten. Darüber hinaus werden die Seiten der Dokumente zufällig getauscht, um dem Modell anzutrainieren, Informationen unabhängig von der Position zu extrahieren.

Die Parameteroptimierung erfolgt iterativ, wie im blau markierten Bereich in Abbildung 3.1 zu erkennen ist, und es werden alle Komponenten des VLM trainiert. Die wichtigsten Parameter der finalen Konfiguration sind in Listing 13 abgebildet.

Um ein Overfitting zu vermeiden, wird neben dem Trainingsdatensatz auch hier der Validierungsdatensatz 3.1.3 verwendet. Nach 80 Dokumenten wird mit dem Validierungsdatensatz der Trainingsfortschritt gemessen. Am Ende des Trainingslaufs wird das Modell geladen, das den geringsten Loss gegenüber dem Validierungsdatensatz hat.

---

<pre> from unsloth import FastVisionModel  model = FastVisionModel.get_peft_model(     model,     finetune_vision_layers=True,     finetune_language_layers=True,     finetune_attention_modules=True,     finetune_mlp_modules=True,     r=16,     lora_alpha=16,     lora_dropout=0,     bias="none",     use_rslora=True,     loftq_config=None, ) </pre>	<pre> from trl import SFTConfig, SFTTrainer  args = SFTConfig(     eval_strategy="steps",     eval_steps=10,     load_best_model_at_end=True,     metric_for_best_model="eval_loss",     neftune_noise_alpha=15,     warmup_steps=15,     num_train_epochs=2,     learning_rate=5e-5,     optim="adamw_8bit",     lr_scheduler_type="cosine",     max_seq_length=20000, ) </pre>
--	--

---

Listing 13: Konfiguration des Fine-Tunings

Abschließend werden wie bei der Auswahl des Basismodells mit dem Testdatensatz 3.1.3 die finalen Metriken bestimmt. Die trainierten LoRA-Adapter werden zur Laufzeit geladen, um Speicherplatz zu sparen, da sonst das Qwen-2.5-VL-7B zusammen mit jedem Adapter gespeichert werden müsste.



# Kapitel 4

## Ergebnisse

Die folgenden Ergebnisse wurden mit dem Testdatensatz (siehe Abschnitt 3.1.3) evaluiert. Bei der Information Extraction wurden die Halluzinationen von Feldern nicht bestraft, da diese durch das Schema gefiltert werden können.

### 4.1 Ergebnisse der OCR/YOLO-Pipeline gegenüber den Basismodellen

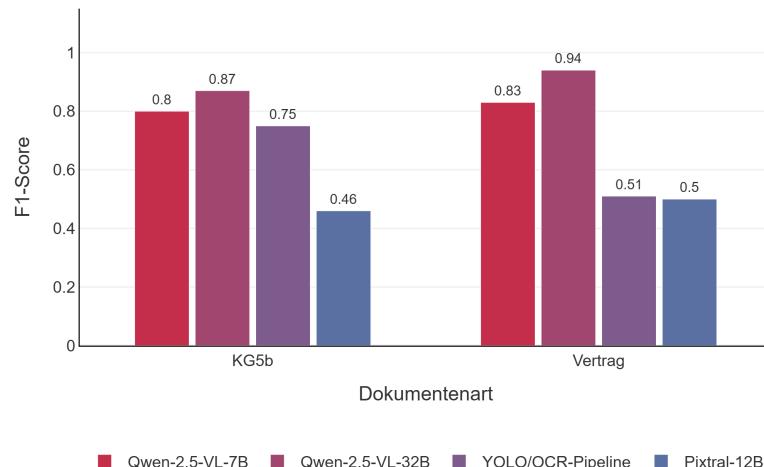


Abbildung 4.1: Ergebnisse der Information Extraction

Abbildung 4.1 stellt die F1-Scores der Basismodelle im Vergleich zur OCR/YOLO-Pipeline für die KG5b-Formulare und Ausbildungsverträge dar.

Bei den KG5b-Formularen zeigt sich ein enges Feld der Qwen-Modelle und der OCR/YOLO-Pipeline. Die Pipeline liegt mit einem F1-Score von 0,75 geringfügig hinter dem Qwen-

2.5-VL-7B. Das größere Qwen-2.5-VL-32B bildet die Spitze mit einem F1-Score von 0,87, während sich das Pixtral-12B am unteren Ende befindet.

Ein differenzierteres Bild zeigt sich bei den heterogenen Verträgen. Die OCR/YOLO-Pipeline fällt auf einen F1-Score von 0,51 ab. Demgegenüber stehen die Qwen-Modelle mit 0,94 (Qwen-2.5-VL-32B) und 0,84 (Qwen-2.5-VL-7B), die bei den Verträgen besser abschneiden als bei den KG5b-Formularen. Das Pixtral-12B bildet erneut das Schlusslicht, liegt jedoch nur minimal hinter der OCR/YOLO-Pipeline.

Die Ergebnisse der Information Extraction zeigen die Abhängigkeit der Leistungsfähigkeit von der Dokumentenart bei der OCR/YOLO-Pipeline. Im Vergleich zu den Qwen-Modellen sinkt die Leistung bei den variablen Verträgen gegenüber den starren KG5b-Formularen. Die Daten belegen zusätzlich, dass bereits ein kleines Modell wie das Qwen-2.5-VL-7B die Pipeline übertrifft. In keiner Dokumentenart schneidet das Pixtral-12B besser als die bisherige Lösung ab.

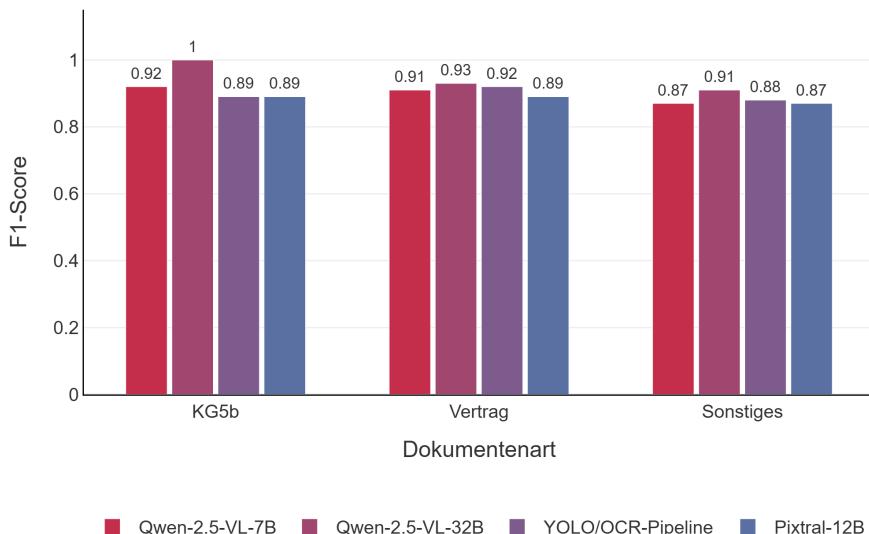


Abbildung 4.2: Ergebnisse der Klassifikation

Abbildung 4.2 visualisiert die Klassifikationsleistung der Basismodelle und der OCR/YOLO-Pipeline anhand von F1-Scores.

Im Gegensatz zur Information Extraction zeigt sich hier ein homogenes Bild. Das Qwen-2.5-VL-32B erzielt in allen drei Kategorien die besten Ergebnisse und klassifiziert die KG5b-Formulare sogar mit einem Score von 1,0. Die Ergebnisse des Pixtral-12B, des Qwen-2.5-VL-7B und der OCR/YOLO-Pipeline liegen über alle Dokumentarten hinweg dicht beisammen. In der Kategorie der sonstigen Dokumente fällt die Leistung aller Modelle gegenüber

den anderen Dokumentarten ab. Hier erzielt das Pixtral-12B zusammen mit dem Qwen-2.5-VL-7B den niedrigsten Score mit 0,87.

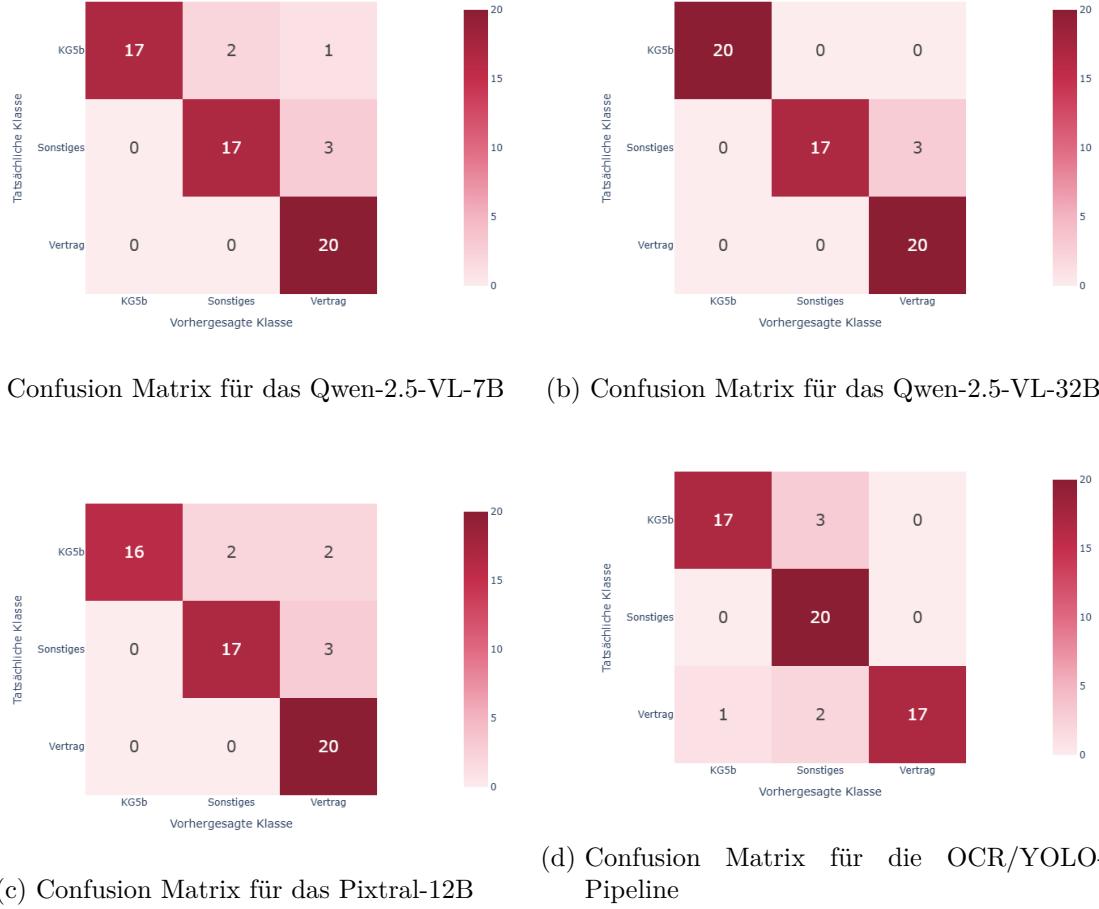


Abbildung 4.3: Confusion Matrices aller Modelle

Die Confusion Matrices in Abbildung 4.3 repräsentieren die Ergebnisse der Klassifikation im Detail.

Während die F1-Scores der Klassifikation ein ähnliches Leistungsniveau zeigen, offenbaren die Matrizen unterschiedliche Fehlerschwerpunkte. Das Qwen-2.5-VL-32B klassifiziert sowohl die KG5b-Formulare als auch die Verträge fehlerfrei. Im Gegensatz dazu erkennt die OCR/YOLO-Pipeline die sonstigen Dokumente perfekt, ordnet diesen aber zusätzlich fünf Dokumente anderer Arten zu. Das Qwen-2.5-VL-7B und das Pixtral-12B haben ein ähnliches Fehlerschema und erkennen die Verträge perfekt. Allerdings klassifizieren beide Modelle KG5b-Formulare sowie sonstige Dokumente häufig als Vertrag.

Die gemeinsame Betrachtung der F1-Scores und der Confusion Matrices verdeutlicht, dass trotz quantitativ vergleichbarer Ergebnisse die Modelle qualitative Unterschiede aufweisen.

Besonders die Stabilität des Qwen-2.5-VL-32B ist hervorzuheben, da die relevanten Dokumente fehlerfrei erkannt werden. Demgegenüber stehen die kleineren VLMs, die Probleme mit der Trennung der Dokumentenklassen haben. Die OCR/YOLO-Pipeline neigt zur Zuordnung zu den sonstigen Dokumenten, wodurch relevante Dokumente verloren gehen.

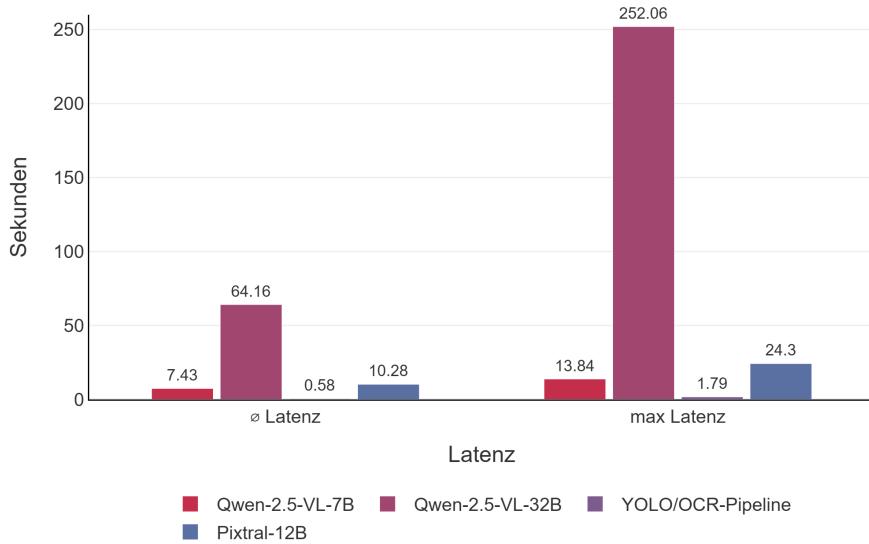


Abbildung 4.4: Ergebnisse der Latenzmessung

Das Diagramm 4.4 zeigt die durchschnittliche und maximale Latenz der Basismodelle und der OCR/YOLO-Pipeline in Sekunden.

Aufgrund der Architektur antwortet die OCR/YOLO-Pipeline mit einer durchschnittlichen Latenz von 0,58 Sekunden nahezu in Echtzeit. Gegenüber der Pipeline benötigt selbst das kleinste Modell fast dreizehnmal länger. Besonders auffällig ist die hohe maximale Latenz des Qwen-2.5-VL-32B mit 252 Sekunden, womit dieses Modell deutlich über den anderen liegt.

Die Latenzmessung belegt den Zusammenhang zwischen der Modellgröße von VLMs und deren Antwortgeschwindigkeit. Mit zunehmender Parameteranzahl wächst die Latenz überproportional.

Diese Ergebnisse begründen zudem die Auswahl des Basismodells. Durch die Überlegenheit des Qwen-2.5-VL-7B gegenüber dem Pixtral-12B in der Information Extraction sowie der Klassifikation stellt sich das Qwen-Modell als besserer Kandidat für das Training heraus.

## 4.2 Ergebnisse des weitertrainierten Modells gegenüber dem größeren Basismodell

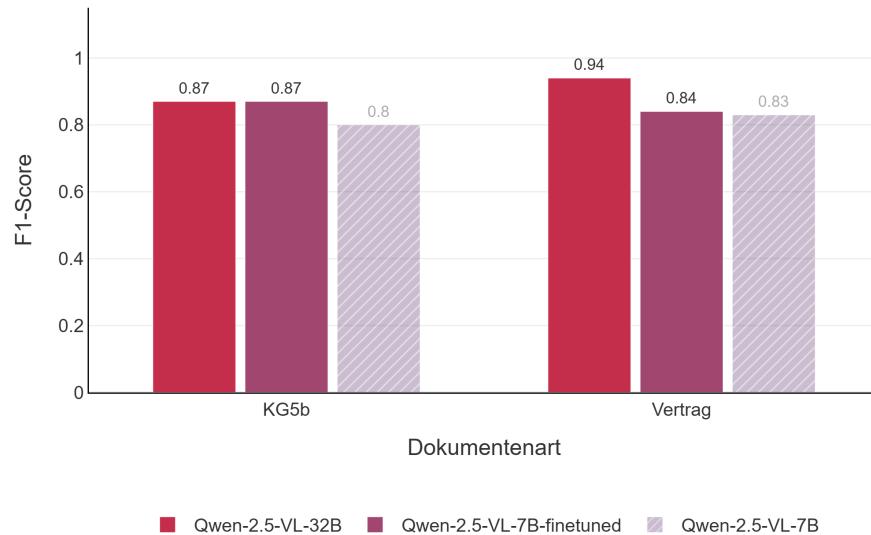


Abbildung 4.5: Ergebnisse der Information Extraction

Abbildung 4.5 stellt die F1-Scores der Information Extraction des Qwen-2.5-VL-7B-finetuned und des Qwen-2.5-VL-32B dar.

Bei den standardisierten KG5b-Formularen erzielen beide Modelle identische Ergebnisse mit einem F1-Score von 0,87. Demgegenüber zeigen die Ausbildungsverträge einen deutlicheren Unterschied. Während das Qwen-2.5-VL-32B einen F1-Score von 0,94 aufweist, liegt das angepasste Modell mit einer Differenz von 0,10 darunter.

Die Verteilung deutet darauf hin, dass ein domänen spezifisches Training ausreicht, um bei starren Layouts auf die Leistung des Qwen-2.5-VL-32B aufzuschließen. Bei variablen Dokumenten wie den Ausbildungsverträgen erzielt das 32B-Modell hingegen weiterhin die höchste Präzision in der Information Extraction.

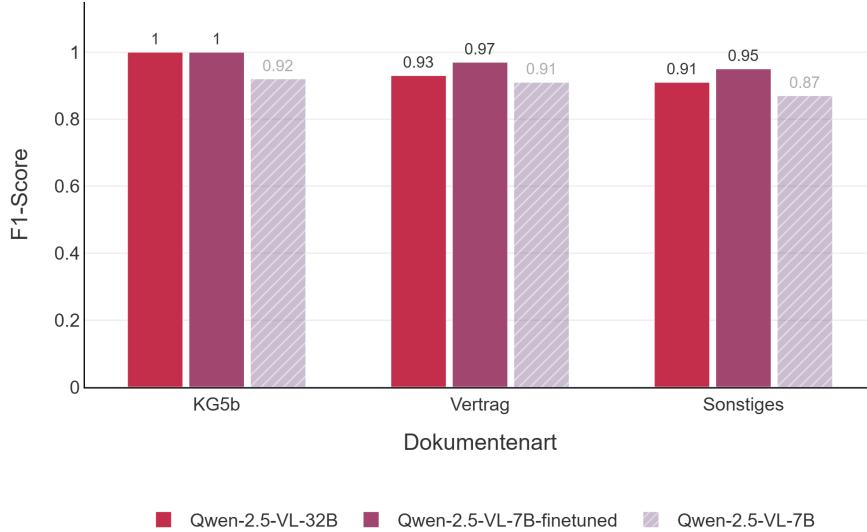


Abbildung 4.6: Ergebnisse der Klassifikation

Abbildung 4.6 visualisiert die Klassifikationsleistung der Modelle anhand der F1-Scores.

Beide Modelle erreichen bei den KG5b-Formularen ein perfektes Ergebnis. Die anderen beiden Kategorien zeigen einen minimalen Vorsprung des angepassten Modells. Hier erreicht das Qwen-2.5-VL-7B-finetuned einen F1-Score von 0,97 bei den Verträgen und 0,95 bei den sonstigen Dokumenten.

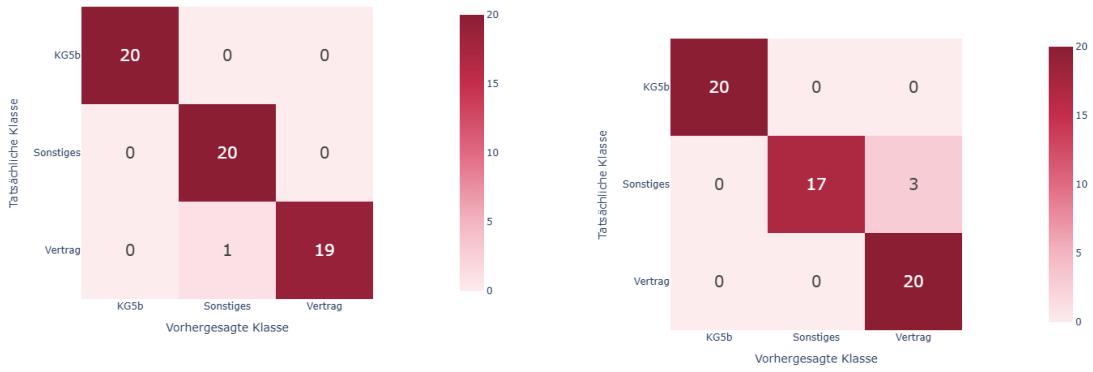


Abbildung 4.7: Confusion Matrices aller Modelle

Einen Einblick in die Fehlerverteilung liefern die Confusion Matrices in Abbildung 4.7.

Das Qwen-2.5-VL-7B-finetuned klassifiziert die KG5b-Formulare sowie die sonstigen Dokumente vollständig korrekt und ordnet lediglich einen Vertrag als sonstiges Dokument ein. Im Gegensatz dazu erkennt das Qwen-2.5-VL-32B die KG5b-Formulare und Verträge fehlerfrei, ordnet jedoch drei der sonstigen Dokumente fälschlicherweise den Verträgen zu.

Die gemeinsame Betrachtung der F1-Scores und der Confusion Matrices belegt, dass das Fine-Tuning die Klassifikationsleistung steigert. Ein spezifisches Training schafft in der Klassifikation einen Vorteil gegenüber einem größeren Basismodell.

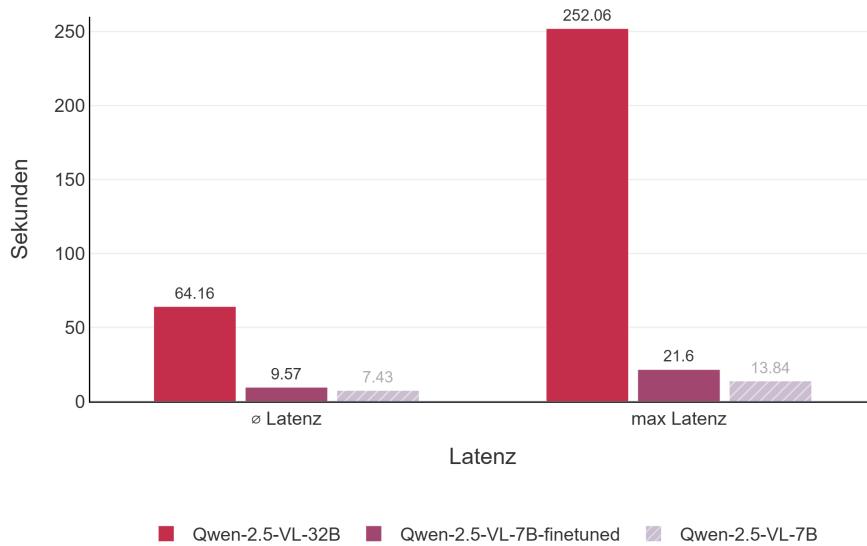


Abbildung 4.8: Ergebnisse der Latenzmessung pro Dokument

Abbildung 4.8 stellt die durchschnittlichen und maximalen Latenzen des Qwen-2.5-VL-32B und Qwen-2.5-VL-7B-finetuned dar.

Es zeigt sich ein deutlicher Vorteil des kleineren Modells. Das Qwen-2.5-VL-7B-finetuned antwortet nach durchschnittlich 9,57 Sekunden, während das Qwen-2.5-VL-32B mehr als das Sechsfache an Zeit benötigt. Bei den Maximalwerten liegt das große Modell mit 252,06 Sekunden deutlich über dem Qwen-2.5-VL-7B-finetuned. Dort liegt die maximale Latenz näher an dem Durchschnittswert.

Die Messwerte verdeutlichen, dass ein Fine-Tuning keinen deutlichen Einfluss auf die Inferenzzzeit gegenüber dem Basismodell hat. Mit der höheren Parameteranzahl benötigt das Qwen-2.5-VL-32B länger zum Antworten als das angepasste Modell.

### 4.3 Ressourcenverbrauch der VLMs gegenüber der OCR/YOLO-Pipeline

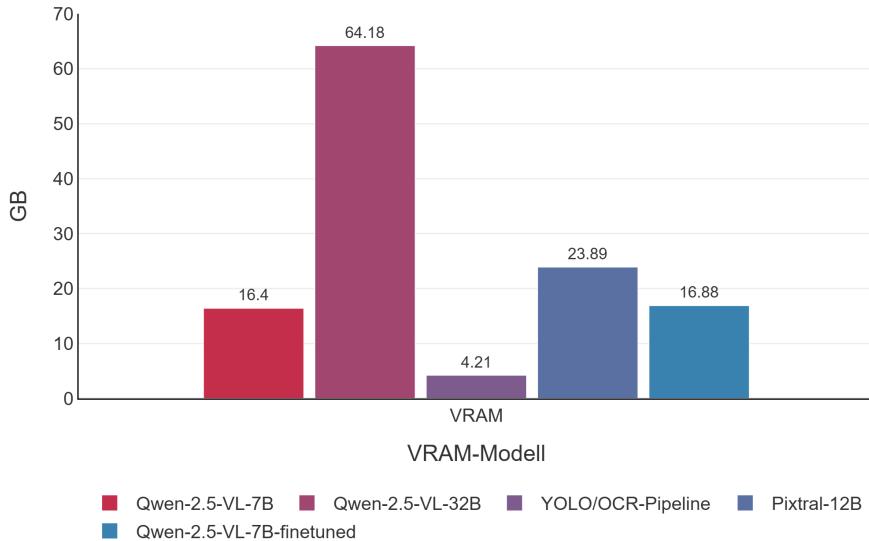


Abbildung 4.9: Ergebnisse der VRAM-Nutzung des geladenen Modells

Abbildung 4.9 visualisiert die statische VRAM-Nutzung der VLMs sowie der OCR/YOLO-Pipeline bei der Initialisierung in Gigabyte.

Die OCR/YOLO-Pipeline verzeichnet trotz der Verwendung mehrerer Modelle einen geringen statischen Verbrauch von 4,21 GB. Die beiden 7B-Modelle liegen auf einem ähnlichen Niveau. Den mit Abstand höchsten Verbrauch weist das Qwen-2.5-VL-32B auf, das mit 64,18 GB den VRAM-Verbrauch der Pipeline um mehr als das Fünfzehnfache übersteigt. Zwischen den drei Qwen-Modellen reiht sich das Pixtral-12B ein.

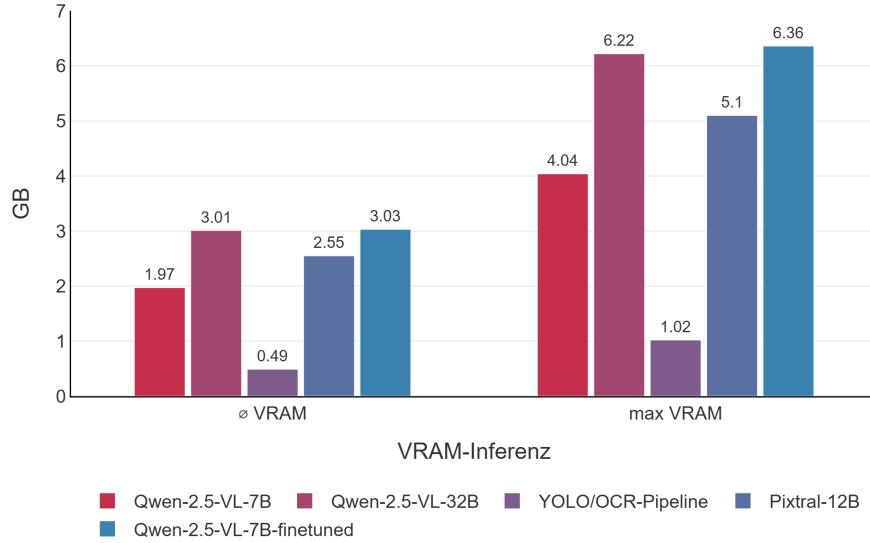


Abbildung 4.10: Ergebnisse der VRAM-Nutzung während der Inferenz pro Dokument

Abbildung 4.10 repräsentiert den durchschnittlichen und maximalen dynamischen Verbrauch in Gigabyte pro Inferenz.

Auch in dieser Messung befindet sich die OCR/YOLO-Pipeline mit einer durchschnittlichen Auslastung von 0,49 GB unter den anderen Modellen. Der maximale Bedarf liegt mit 1,02 GB unter dem Durchschnittsverbrauch des effizientesten VLMs. Auffällig ist das Verhältnis des Qwen-2.5-VL-7B-finetuned und des größeren Qwen-2.5-VL-32B. Das kleinere, trainierte Modell hat mit durchschnittlich 3,03 GB einen geringfügig höheren Verbrauch als das 32B-Modell.

Die gemeinsame Betrachtung der statischen und dynamischen VRAM-Auslastung verdeutlicht den Architekturunterschied zwischen der Pipeline und den VLMs. Selbst das kleinste Modell ist im Gegensatz zur OCR/YOLO-Pipeline um ein Vielfaches größer, was einen deutlichen Unterschied bezüglich der benötigten Hardware zur Laufzeit darstellt.

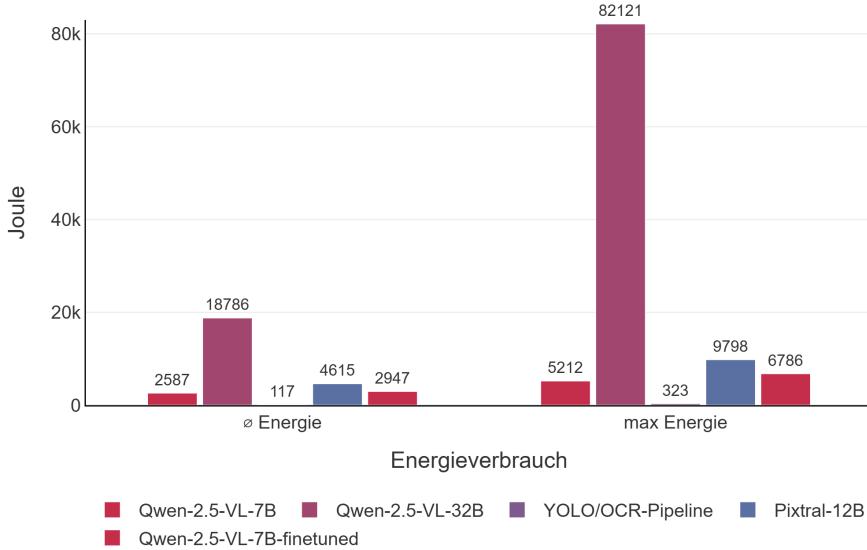


Abbildung 4.11: Ergebnisse des Energieverbrauchs pro Dokument

Abbildung 4.11 veranschaulicht den durchschnittlichen und maximalen Energieverbrauch pro Inferenz in Joule.

Der Energieverbrauch zeigt den markantesten Unterschied zwischen den Systemen. Die OCR/YOLO-Pipeline verbraucht durchschnittlich lediglich 117 Joule. Selbst das effizienteste Modell, das Qwen-2.5-VL-7B, verbraucht durchschnittlich das 22-Fache der Pipeline. Das Qwen-2.5-VL-7B-finetuned liegt knapp über seinem Basismodell. Den mit Abstand höchsten Energiebedarf hat das Qwen-2.5-VL-32B. Mit durchschnittlich 18.786 Joule und maximal 82.121 Joule übertrifft das Modell die Pipeline im Mittel um das 160-Fache.

Die Messung des Energieverbrauchs belegt den quantitativen Unterschied der VLMs gegenüber der spezialisierten Pipeline. Es wird deutlich, dass die Skalierung der Parameteranzahl mit einem höheren Energieverbrauch einhergeht. Obwohl die VLMs eine höhere Klassifikations- und Extraktionsleistung aufweisen, verdeutlicht der reine Energieverbrauch, dass der Einsatz eines solchen Modells mit einem ungleich höheren Ressourceneinsatz einhergeht.

# Kapitel 5

## Diskussion

### 5.1 Interpretation der Fehler

Sowohl die OCR/YOLO-Pipeline als auch die VLMs weisen spezifische Fehlermuster auf, die sich in Klassifikations-, Format-, Detektions- und Wertfehler unterteilen lassen.

Klassifikationsfehler wiegen in der aus mehreren Komponenten bestehenden OCR/YOLO-Pipeline besonders schwer, da die Klassifikation hier einen logischen Single Point of Failure darstellt. Wie die Confusion Matrix in Abbildung 4.3 verdeutlicht, wurden fünf Dokumente fälschlicherweise als **Sonstiges** eingestuft, was aufgrund der Systemarchitektur (siehe Abbildung 2.3) zu einem sofortigen Abbruch der Verarbeitung führt. Bei diesen fünf Dokumenten handelte es sich auffälligerweise ausschließlich um fotografierte Dokumente. Fotos weisen häufig Verzerrungen, Schattenwurf oder eine schlechte Beleuchtung auf, wodurch die OCR-Ergebnisse des vorgeschalteten Systems schlechter ausfallen und die Klassifikation erschweren.

Diese Klassifikationsfehler mindern die Leistung der Information Extraction, da mit einem einzigen Fehler bereits zehn (Vertrag) oder zwölf (KG5b) Felder fehlen. Auch die Verwechslung von Verträgen mit KG5b-Formularen wirkt sich negativ auf die OCR/YOLO-Pipeline aus, da für beide Dokumentarten unterschiedliche, spezialisierte Modelle trainiert wurden. Ein entscheidender Vorteil der generalistischen VLMs ist hier die Robustheit durch das einheitliche Modell. Da die Felder eines Vertrags eine Teilmenge des KG5b-Schemas darstellen, bleibt die Information Extraction teilweise erfolgreich, selbst wenn das Dokument falsch klassifiziert wurde. Ein Fehler in der Klassifikation wirkt sich somit nicht so stark aus wie bei der OCR/YOLO-Pipeline. Im Idealfall fehlt bei der Erkennung eines Vertrages statt eines KG5b-Formulars kein Feld und im umgekehrten Fall fehlen maximal zwei Felder. Die Fehlklassifizierung als **Sonstiges** ist jedoch auch bei den VLMs kritisch, da das Modell in diesem Fall keine Extraktion durchführt. Wie in den Confusion Matrices in Abbildung 4.7 ersichtlich, erweisen sich jedoch insbesondere das Qwen-2.5-VL-7B-finetuned und das Qwen-2.5-VL-32B als äußerst zuverlässig in der Klassifizierung.

Die Detektionsfehler beziehen sich primär auf Checkboxen, Stempel und Unterschriften. Checkboxen sind im KG5b-Formular für das Feld `apprenticeship_finished` relevant (siehe Abbildung 2.1a). Hier müssen die Modelle zwischen leeren, angekreuzten und durchgestrichenen Boxen unterscheiden. Detektionsfehler sind häufig auf den unvermeidbaren Medienbruch zurückzuführen, der entsteht, wenn Dokumente zur Unterschrift ausgedruckt und anschließend wieder digitalisiert werden. Durch diesen Prozess verlieren Stempel und Unterschriften an Kontrast oder werden in Schwarz-Weiß-Scans schwer erkennbar.

Zusätzlich erschwert die räumliche Nähe der Felder `date_document` und `signature_company` die Extraktion. Da handschriftliche Unterschriften regelmäßig den vorgesehenen Platz überschreiten, wird das Datumsfeld häufig überdeckt. Während die OCR/YOLO-Pipeline hier oft keine valide Bounding-Box mehr liefern kann, zeigen das Qwen-2.5-VL-32B und das Qwen-2.5-VL-7B-finetuned eine hohe Robustheit gegenüber diesen Überlagerungen. Die trotz der Überlappungen gute Leistung ist vermutlich auf die Augmentation der Daten im ursprünglichen Training zurückzuführen.

Formatfehler verdeutlichen Unterschiede in den *Instruction-Following*-Fähigkeiten der Modelle. Während die Qwen-Modelle nahezu fehlerfrei valide JSON-Strukturen lieferten, scheiterte das Pixtral-12B trotz Retry-Logik an zwei Verträgen. Das Modell generierte in keinem der Versuche die umschließenden Klammern ({}), wodurch das Parsen fehlschlug und 20 JSON-Felder verloren gingen. Dies deutet darauf hin, dass das Pixtral-12B eine geringere Stabilität bei der Einhaltung strikter Ausgabeformate aufweist als die Qwen-Modelle.

Bei den Wertfehlern muss zwischen OCR-Fehlern und Mapping-Fehlern (falsche semantische Zuordnung) unterschieden werden. Die OCR/YOLO-Pipeline zeigte Schwächen in der Erkennung von Namen und Datumsangaben. Selbst die Levenshtein-Distanz mit einem Schwellenwert von 0,8 reichte in einigen Fällen nicht aus, um diese Fehler zu kompensieren. Beim Pixtral-12B und Qwen-2.5-VL-7B traten neben OCR-Fehlern vermehrt Mapping-Fehler auf, bei denen beispielsweise das Geburtsdatum des Kindes mit dem Startdatum der Ausbildung vertauscht wurde. Das Fine-Tuning des Qwen-2.5-VL-7B reduzierte sowohl die OCR- als auch die Mapping-Fehler spürbar.

Halluzinationen von Feldern traten primär bei den Basismodellen auf, insbesondere bei der Klasse **Sonstiges**, da die Modelle hier Felder der beiden anderen Schemata extrahierten. Da diese zusätzlichen Felder durch die Schemata gefiltert werden können, stellen sie, anders als inhaltliche Halluzinationen, keine Einschränkung dar.

## 5.2 Abwägung Effektivität und Ressourceneffizienz

Die Ergebnisse der Untersuchung offenbaren einen Zielkonflikt zwischen der Leistungsfähigkeit der Modelle und ihrem Ressourcenbedarf. Während die OCR/YOLO-Pipeline durch

ihre extrem geringe Latenz und Ressourceneffizienz auffällt, zeigen die VLMs eine Überlegenheit in der Qualität der Information Extraction.

Die OCR/YOLO-Pipeline bildet mit einem durchschnittlichen Energieverbrauch von 117 Joule und einer Latenz von unter einer Sekunde die Basislinie für die Effizienz. Demgegenüber steht das leistungsstärkste Modell, das Qwen-2.5-VL-32B, welches im Mittel die 160-fache Energie pro Dokument benötigt und eine Latenz von durchschnittlich 64,16 Sekunden aufweist. Selbst das kleinste Modell, das Qwen-2.5-VL-7B, verbraucht rund das 22-fache an Energie. Kritisch betrachtet relativiert sich der reine Effizienzvorteil der Pipeline jedoch im operativen Kontext. Ein niedriger F1-Score (0,51 bei Verträgen) zieht eine manuelle Nachbearbeitung durch Sachbearbeiter nach sich. Die durch die schnelle technische Verarbeitung gewonnene Zeit wird somit im Gesamtprozess durch menschliche Korrektureingriffe wieder aufgezehrt.

Betrachtet man die Wirtschaftlichkeit, so übersteigen die Kosten für eine einminütige manuelle Korrektur die Energiekosten einer einminütigen GPU-Inferenz um ein Vielfaches. Die höhere Rechenzeit der VLMs gleicht sich somit durch die Einsparung von Arbeitszeit in der Sachbearbeitung aus. Bei einem F1-Score bei Verträgen von 0,94 (Qwen-2.5-VL-32B) muss im Schnitt ein Feld korrigiert werden, während bei einem F1-Score von 0,51 (OCR/YOLO-Pipeline) vier bis fünf Felder korrigiert werden müssen.

Neben dem Energiebedarf ist die Infrastruktur, spezifisch die VRAM-Auslastung, ein entscheidender Faktor. Während die OCR/YOLO-Pipeline auf einfachen GPUs lauffähig ist, setzen die großen VLMs leistungsstarke GPUs voraus. Das Qwen-2.5-VL-32B übertrifft das Qwen-2.5-VL-7B-finetuned zwar in der Information Extraction bei Verträgen (F1-Score 0,94 gegenüber 0,84), erkauft sich diesen Vorteil jedoch mit einem wesentlich höheren VRAM-Bedarf.

Insgesamt bietet das Qwen-2.5-VL-7B-finetuned den besten Kompromiss zwischen Effizienz und Effektivität. Das trainierte Modell schlägt dabei in der Klassifikation das Qwen-2.5-VL-32B, nähert sich bei der Information Extraction der Leistung des großen Modells an und ist dabei effizienter. Mit einem statischen VRAM-Bedarf von circa 17 GB ermöglicht dieses Modell den Betrieb auf kleineren GPUs, was die Hürden für eine Integration in die Infrastruktur senkt. Hinsichtlich der Skalierbarkeit stellt die Latenz der VLMs jedoch weiterhin eine Herausforderung dar, da eine niedrige Latenz ein Cluster aus GPUs voraussetzt.

### 5.3 Komplexität und Wartbarkeit der Systemarchitekturen

Neben der Modellleistung stellt die Komplexität der Infrastruktur einen zentralen Faktor für den produktiven Einsatz dar. Zur Einordnung wird das Qualitätsmodell der ISO/IEC 25010

herangezogen, wobei der Fokus auf dem Qualitätsmerkmal der Wartbarkeit mit den Teilaспектen Modularität, Analysierbarkeit, Änderbarkeit und Testbarkeit liegt.

Die Architektur der OCR/YOLO-Pipeline zeichnet sich durch eine Verkettung spezialisierter Komponenten aus (siehe Abbildung 2.3). Obwohl dieser Aufbau grundsätzlich modular ist, entsteht durch die sequenzielle Verarbeitung eine kritische Abhängigkeit. Der Klassifikator nimmt eine zentrale Rolle ein, da seine Entscheidung den weiteren Verlauf bestimmt. Fehlklassifikationen pflanzen sich durch die nachgelagerten Komponenten fort, was die Fehleranalyse erschwert. Wie bereits in der Fehleranalyse 5.1 identifiziert, stellt der Klassifikator hier architektonisch einen *Single Point of Failure* dar.

Hinsichtlich der Änderbarkeit offenbart die OCR/YOLO-Pipeline einen erhöhten Aufwand. Die Einführung einer neuen Dokumentenart erfordert nicht nur Anpassungen am Klassifikator, sondern auch das Training eines neuen YOLO-Modells. Dies ist mit hohem manuellem Aufwand verbunden, da das Annotieren von Bounding-Boxes zeitintensiv ist. Des Weiteren gibt es durch die vielen Komponenten zahlreiche Schnittstellen, die bei einer Änderung der Dokumentenstrukturen gegebenenfalls angepasst werden müssen. Im Gegensatz dazu reduziert der generalistische Ansatz mit den VLMs die Anzahl der Komponenten und Schnittstellen. Änderungen an den Dokumentarten beschränken sich auf die Anpassung des Prompts oder, im Fall des Qwen-2.5-VL-7B-finetuned, auf das Training eines Modells. Hierbei muss ebenfalls ein Modell trainiert und gegebenenfalls die Schnittstelle angepasst werden, aber die Annotation der Daten ist aufgrund von Model-Assisted-Labeling wesentlich einfacher.

Der monolithische Ansatz bringt jedoch Nachteile für die Testbarkeit mit sich. Ein VLM agiert als Black Box, wodurch die interne Logik schwer analysierbar ist. Während die Komponenten in der OCR/YOLO-Pipeline logisch entkoppelt sind und eine Änderung am YOLO-Modell das OCR-Modell unberührt lässt, wirken sich Änderungen bei einem VLM auf alle Dokumentarten aus. Eine Änderung des Prompts oder das Fine-Tuning zur Verbesserung einer Dokumentenart kann zu Regressionen bei anderen Dokumentenarten führen. Dies erhöht die Komplexität von Regressionstests. Mit der Einführung von automatisierten Tests anhand des Validierungsdatensatzes nach Änderungen kann dieses Risiko jedoch minimiert werden.

## 5.4 Zusammenfassende Beantwortung der Forschungsfragen

### 5.4.1 Beantwortung der Forschungsfrage 1

1. Wie verhält sich ein nicht domänenpezifisch angepasstes VLM hinsichtlich Latenz, Klassifikations- und Extraktionsleistung im Vergleich zur OCR/YOLO-Pipeline, und

welche Vorteile bietet ein einzelnes generalistisches Modell gegenüber spezialisierten Einzelsystemen?

Hinsichtlich der Latenz zeigt ein nicht domänenspezifisch angepasstes VLM Nachteile gegenüber der OCR/YOLO-Pipeline. Selbst das effizienteste Basismodell, das Qwen-2.5-VL-7B, benötigt mit durchschnittlich 7,43 Sekunden fast 13-mal so viel Zeit wie die bisherige Lösung.

Während die Klassifikationsleistung der Basismodelle mit der Leistung der Pipeline übereinstimmt, zeigt sich ein klarer Unterschied in der Extraktionsgüte. Bei Verträgen erreicht die OCR/YOLO-Pipeline im Gegensatz zu den Qwen-Modellen lediglich einen F1-Score von 0,51. Mit einem F1-Score von 0,83 erreicht selbst das kleinste Basismodell, das Qwen-2.5-VL-7B, eine Leistungssteigerung von 0,32 Punkten. Auch bei den starren KG5b-Formularen übertrifft das Qwen-2.5-VL-7B die Pipeline knapp. Einzig das Pixtral-12B erzielte gegenüber der Pipeline sowohl in der Klassifikation als auch in der Information Extraction schlechtere Ergebnisse.

Neben der verbesserten Leistung bietet ein generalistischer Ansatz wesentliche Vorteile in Bezug auf Architektur und Wartbarkeit. Durch den Single Point of Failure des Klassifikators ist die OCR/YOLO-Pipeline weniger robust gegenüber Klassifikationsfehlern. Eine Fehlklassifikation führt zur Verwendung falscher Modelle oder zum sofortigen Abbruch. Ein VLM hingegen ist weniger anfällig bei einem Teil von Fehlklassifikationen, da aufgrund überlappender Schemata dennoch relevante Informationen extrahiert werden.

Der monolithische Ansatz eines VLM reduziert des Weiteren die Anzahl der Komponenten und Schnittstellen drastisch. Während die Pipeline für jede neue Dokumentenart neue Modelle und Schnittstellen benötigt, reduziert sich der Aufwand bei einem VLM. Anstatt zeitaufwendig Bounding-Boxes für das Training von YOLO-Modellen zu annotieren, genügt die Anpassung des Prompts oder ein Fine-Tuning mittels Model-Assisted-Labeling, was den manuellen Aufwand senkt.

#### 5.4.2 Beantwortung der Forschungsfrage 2

2. Welche Auswirkungen hat ein domänenspezifisches Training eines kleineren Modells auf dessen Latenz, Klassifikations- und Extraktionsleistung im Vergleich zu einem leistungsstärkeren Basismodell?

Der Vergleich zeigt, dass das Training eines kleineren Modells einen merklichen Effizienzvorteil gegenüber einem größeren Basismodell bringt und in der Klassifikation sogar überlegen ist, wobei jedoch leichte Abstriche bei der Information Extraction hingenommen werden müssen.

Hinsichtlich der Latenz schlägt das angepasste Modell aufgrund seiner Größe das 32B-Modell deutlich. Insbesondere die maximale Latenz des großen Modells ist mit 252,06 Sekunden bedenklich.

Die Klassifikationsleistung ist bei beiden Modellen beeindruckend, jedoch schneidet hier das angepasste Modell besser ab. Mit einem F1-Score von 0,95 bei sonstigen Dokumenten erzielt das Qwen-2.5-VL-7B-finetuned einen minimal höheren Wert als das Qwen-2.5-VL-32B mit einem F1-Score von 0,91. Dies verdeutlicht, dass das kleinere Modell durch das Training besser zwischen den Dokumentarten unterscheiden kann als das Qwen-2.5-VL-32B mit seinem breiten Vorwissen.

Bei der Information Extraction hängt die Leistung von der Varianz der Dokumentarten ab. Bei den starren KG5b-Formularen erzielen beide Modelle einen F1-Score von 0,87. Die durch hohe Varianz gekennzeichneten Ausbildungsverträge werden jedoch durch das Qwen-2.5-VL-32B mit einem F1-Score von 0,94 besser erkannt. Das Weltwissen des großen Modells hilft ihm dabei, viele verschiedene Varianten von Dokumenten besser zu verstehen.

Trotz der minimal besseren Information Extraction des großen Modells bietet das Qwen-2.5-VL-7B-finetuned den besten Kompromiss aus Effizienz und Effektivität.

#### 5.4.3 Beantwortung der Forschungsfrage 3

3. Wie unterscheidet sich der Ressourcenverbrauch (Energieverbrauch und VRAM-Auslastung) der VLMs von dem der OCR/YOLO-Pipeline?

Der Vergleich des Ressourcenverbrauchs zeigt die gravierendsten Unterschiede zwischen der bisherigen Pipeline und den VLMs.

Beim Energieverbrauch verzeichnet die Pipeline mit 117 Joule im Durchschnitt pro Dokument einen deutlichen Unterschied zum Verbrauch der VLMs. Selbst das effizienteste Modell, das Qwen-2.5-VL-7B, steht mit einem Verbrauch von durchschnittlich 2587 Joule pro Dokument in starkem Kontrast dazu. Das leistungsstarke Qwen-2.5-VL-32B verbraucht mit im Mittel 18.786 Joule sogar das 160-fache der Energie der Pipeline.

Ein kritischer Aspekt für den operativen Betrieb ist neben dem statischen auch der dynamische VRAM-Verbrauch. Während die OCR/YOLO-Pipeline mit 0,49 GB kaum ins Gewicht fällt und der Speicher durch die geringe Latenz schnell wieder freigegeben wird, verbrauchen die VLMs beträchtlich mehr. Interessanterweise liegt der durchschnittliche dynamische Verbrauch des Qwen-2.5-VL-7B-finetuned mit 3,03 GB leicht über dem des größeren Qwen-2.5-VL-32B (3,01 GB). Trotz des nahezu gleichen dynamischen VRAM-Verbrauchs belegt das Qwen-2.5-VL-7B-finetuned mit 23,89 GB jedoch weniger statischen VRAM, wodurch mehr Speicher für die dynamische Allokation zur Verfügung steht.

Das Problem des dynamischen Verbrauchs liegt in der Variabilität. Mit jeder Anfrage wächst der Verbrauch, weshalb die GPUs über genügend Puffer verfügen müssen, um Out-of-Memory-Fehler zu vermeiden. Besonders bei der Verarbeitung von mehreren Bildern wird der Kontext pro Anfrage groß. Die Skalierung fällt hier wesentlich schwerer als die der deterministischen OCR/YOLO-Pipeline, da abhängig von der gewünschten Latenz und den gleichzeitigen Anfragen über die benötigte Hardware entschieden werden muss.

## 5.5 Limitationen und Probleme der Evaluation

Im Verlauf der Implementierung und Evaluation traten technische Limitationen und Herausforderungen auf, die die Ergebnisse beeinflussen und Potenziale für zukünftige Optimierungen aufzeigen. Diese lassen sich in framework-spezifische Probleme, Aspekte der Ressourcenverwaltung sowie methodische Grenzen der Klassifikation und Datenverarbeitung unterteilen.

Eine technische Hürde stellte die Inferenz des Pixtral-12B dar. Das Modell wurde über den in Listing 14 dargestellten Code geladen. Hierbei zeigte sich ein Memory Leak, das trotz Bereinigung des Speichers und des Caches dazu führte, dass der VRAM beider Grafikkarten nach vier bis fünf Inferenzen vollständig belegt war. Dieses Verhalten trat nur beim Pixtral-12B auf und zwang zur manuellen Verarbeitung der Datensätze in Batches.

---

```
from mistral_inference.transformer import Transformer
from mistral_inference.generate import generate

from mistral_common.tokenizers.mistral import MistralTokenizer
from mistral_common.protocol.instruct.messages import UserMessage, TextChunk, ImageURLChunk
from mistral_common.protocol.instruct.request import ChatCompletionRequest

tokenizer = MistralTokenizer.from_file(f"{mistral_models_path}/tekken.json")
model = Transformer.from_folder(mistral_models_path)

...
out_tokens, _ = generate([tokens], model, images=[images], max_tokens=1024, temperature=0)
result = tokenizer.decode(out_tokens[0])
```

---

Listing 14: Laden des Pixtral-Modells

Ein weiteres Problem bei der Implementierung trat bei den Qwen-Modellen auf. Anfänglich wurden die Dokumente als Base64-Strings an das Modell übergeben. Die in Listing 15

verwendete Funktion `apply_chat_template` wies jedoch einen internen Fehler bei der Verarbeitung mehrerer Base64-Bilder auf, sodass lediglich das erste Bild vom Modell verarbeitet wurde. Dieses Problem konnte durch die Umstellung auf PIL-Images gelöst werden.

---

```
from transformers import Qwen2_5_VLForConditionalGeneration, AutoTokenizer, AutoProcessor
from qwen_vl_utils import process_vision_info

model = Qwen2_5_VLForConditionalGeneration.from_pretrained(
    "Qwen/Qwen2.5-VL-7B-Instruct", torch_dtype="auto", device_map="auto"
)
processor = AutoProcessor.from_pretrained("Qwen/Qwen2.5-VL-7B-Instruct")

...

text = processor.apply_chat_template(
    messages, tokenize=False, add_generation_prompt=True
)
```

---

Listing 15: Laden der Qwen-Modelle

Hinsichtlich der Ressourceneffizienz zeigten die Ergebnisse in Abbildung 4.10 und Abbildung 4.9, dass das Qwen-2.5-VL-7B-finetuned im Vergleich zum Basismodell einen erhöhten statischen und im Vergleich zum 32B-Modell sogar einen höheren dynamischen VRAM-Verbrauch aufweist. Dieser Overhead resultiert vermutlich aus der Art der Bereitstellung. Die Gewichte der während des Fine-Tunings trainierten LoRA-Adapter werden nicht fest mit dem Basismodell verschmolzen, sondern zur Laufzeit dynamisch geladen, um Speicherplatz zu sparen. In einem produktiven Einsatz wird der finale Adapter mit dem Basismodell verschmolzen und zusammen gespeichert, was diesen Overhead reduzieren wird.

Methodische Limitationen zeigten sich in der Robustheit der Klassifikation bei Fehlklassifikationen. Wie in der Fehleranalyse diskutiert, führt eine Fehlklassifikation eines Vertrags als **Sonstiges** zum Verlust aller Felder, da für die Klasse **Sonstiges** keine Extraktion stattfindet. Das Extrahieren von Vertragsdaten auch aus sonstigen Dokumenten würde die negativen Auswirkungen einer solchen Fehlklassifizierung reduzieren. Jedoch wäre dieses Vorgehen mit einem erhöhten Labeling-Aufwand verbunden.

Zur Steigerung der Zuverlässigkeit der Ausgabeformate bietet sich der Einsatz von Constrained Decoding an. Durch die Bereitstellung des Modells über Frameworks wie vLLM kann die Ausgabe strikt auf ein JSON-Schema beschränkt werden. Damit ließe sich nicht nur die Generierung invalider JSON-Schemata verhindern, sondern auch das Format für Datumsangaben und binäre Felder festlegen. Zusätzlich ermöglicht dieser Ansatz die Auswertung der logprobs der generierten Token. Durch die Analyse der Wahrscheinlichkeitsverteilung des

Tokens, das auf "type: " folgt, ließe sich eine Confidence für die Klassifikation bestimmen. Dokumente, bei denen das Modell eine niedrige Confidence aufweist, könnten so für den Sachbearbeiter markiert werden.

Außerdem wurde in der Evaluation nicht berücksichtigt, dass eventuell mehrere Dokumentarten in einem PDF zusammengefasst sein könnten. Um gemischte Dokumentenarten zu erkennen, müsste das Modell in der Lage sein, ein Array von JSON-Objekten zu liefern. Hierfür müsste eine Anpassung der Trainingsdaten mittels Augmentation erfolgen, da für diese vermischten Dokumente nur wenig Trainingsdaten zur Verfügung stehen.

Eine weitere Limitation stellt die maximale Kontextlänge der Modelle dar. Dokumente, die besonders umfangreich sind, neigen dazu, das Kontextfenster des Modells zu überlasten. Dies führt zu einem Abbruch der Inferenz oder zu einem Abschneiden relevanter Inhalte. Für den Fall derartig vielseitiger Dokumente muss für den produktiven Einsatz eine Strategie, wie beispielsweise eine Vorfilterung, entwickelt werden.

## 5.6 Handlungsempfehlung für die Bundesagentur für Arbeit

Aus der Beantwortung der Forschungsfragen und der vorangegangenen Diskussion leitet sich eine Handlungsempfehlung für die Bundesagentur für Arbeit ab. Die Entscheidung beruht auf einer Abwägung zwischen ökologischem Fußabdruck, Leistungsfähigkeit und wirtschaftlicher Notwendigkeit.

Als primäre Empfehlung wird der Einsatz des Qwen-2.5-VL-7B-finetuned ausgesprochen. Dieses Modell stellt im Vergleich zur bestehenden OCR/YOLO-Pipeline sowie zum größeren Basismodell Qwen-2.5-VL-32B den besten Kompromiss dar. Zwar weist die OCR/YOLO-Pipeline den geringsten ökologischen Fußabdruck auf, jedoch verursacht die geringe Extraktionsgüte bei Verträgen hohe Personalkosten. Die Kosten der manuellen Korrektur durch Sachbearbeiter übersteigen die Kosten des Qwen-2.5-VL-7B-finetuned. Der Einsatz des Qwen-2.5-VL-7B-finetuned erhöht zwar den ökologischen Fußabdruck, bringt aber eine Reduzierung der Bearbeitungszeit pro Antrag mit sich.

Für die Skalierung über mehrere Projekte hinweg wird davon abgeraten, ein einzelnes 32B-Modell bereitzustellen. Stattdessen könnten für die verschiedenen Fachverfahren LoRA-Adapter entwickelt werden, die, wie in der Evaluation, zur Laufzeit geladen werden. Durch die geringere Parameteranzahl könnten drei bis vier Instanzen statt eines Qwen-2.5-VL-32B betrieben werden. Die dynamische Anpassung der Adapter verbraucht minimal mehr Speicher und Energie, ist in der Summe jedoch sparsamer als der Betrieb eines großen Modells.

Zusammenfassend ermöglicht der Wechsel auf das Qwen-2.5-VL-7B-finetuned der Bundesagentur für Arbeit, den Automatisierungsgrad bei Kindergeldanträgen zu erhöhen und das Fachpersonal zu entlasten, ohne dabei unverhältnismäßige Ressourcen zu binden.

# Kapitel 6

## Zusammenfassung und Ausblick

### 6.1 Zusammenfassung

Vor dem Hintergrund des demografischen Wandels und steigender Antragszahlen ist die Automatisierung von Verwaltungsprozessen für die Bundesagentur für Arbeit von Bedeutung. Die vorliegende Bachelorarbeit untersuchte das Potenzial von multimodalen Large Language Models als Nachfolger für die bestehende OCR/YOLO-Pipeline zur Verarbeitung von Ausbildungsnachweisen volljähriger Auszubildender. Ziel war es, die Extraktionsleistung der bisherigen Lösung unter Beobachtung des Ressourcenverbrauchs zu verbessern.

Im Rahmen der Evaluation wurden die Modelle Pixtral-12B, Qwen-2.5-VL-7B und Qwen-2.5-VL-32B gegen die Pipeline getestet. Ein zentraler Bestandteil der Untersuchung war das Fine-Tuning des kleineren 7B-Modells mittels LoRA, um dessen Leistungsfähigkeit gegenüber dem Qwen-2.5-VL-32B zu bewerten.

Die Ergebnisse zeigen, dass die herkömmliche OCR/YOLO-Pipeline bei heterogenen Ausbildungsverträgen an ihre Grenzen stößt (F1-Score von 0,51). Die VLMs konnten hier eine Leistungssteigerung erzielen. Das Qwen-2.5-VL-32B lieferte mit einem F1-Score von 0,94 die besten Ergebnisse, erwies sich jedoch aufgrund einer durchschnittlichen Latenz von 64 Sekunden und eines extrem hohen Energiebedarfs als unwirtschaftlich. Das nachtrainierte Qwen-2.5-VL-7B-finetuned bestätigte, dass domänen spezifisch angepasste, kleine Modelle mit großen Generalisten konkurrieren können. Mit einem F1-Score von 0,84 bei Verträgen und einer Latenz von unter 10 Sekunden bietet es den optimalen Kompromiss. Zwar steigt der Energieverbrauch im Vergleich zur alten Pipeline an, dies steht jedoch der Minderung der manuellen Nachbearbeitung gegenüber.

Als Fazit ergibt sich, dass der Einsatz eines kleinen, angepassten Modells den nächsten logischen Schritt in der teil-automatisierten Verarbeitung von Dokumenten darstellt.

## 6.2 Ausblick

Obwohl der entwickelte Prototyp bereits vielversprechende Ergebnisse liefert, stehen dem Einsatz in einer produktiven Umgebung noch Hürden im Weg. Die Umgebung der bisherigen Pipeline muss für den Einsatz angepasst und die Ausgabe des VLM stabilisiert werden. Durch die Einführung von Constrained Decoding mithilfe von vLLM wird in Zukunft die Generierung der JSON-Objekte sichergestellt. Außerdem ermöglicht das Auslesen von Token-Wahrscheinlichkeiten eine Auswertung der Konfidenz für die Klassifikation.

Hinsichtlich der Robustheit muss die VLM-Pipeline für komplexere Szenarien, wie zum Beispiel vermischte oder vielseitige Dokumente, weiter angepasst werden.

Um das Potenzial kleiner, trainierter Modelle anderen Fachverfahren zu erläutern, wird das Proof of Concept anderen Abteilungen vorgestellt. Dies fördert die weitere Digitalisierung der Bundesagentur für Arbeit.

# Abbildungsverzeichnis

2.1	Exemplarisch ausgefülltes KG5b-Formular mit Markierung der relevanten Felder	4
2.2	Synthetischer Ausbildungsvertrag der Industrie- und Handelskammer mit markierten Datenfeldern	5
2.3	Schematischer Workflow der aktuellen OCR/YOLO-Pipeline zur Dokumentenverarbeitung	5
2.4	Schematische Darstellung der Trainings- und Inferenzinfrastruktur im Kubernetes-Cluster	7
2.5	Architektur eines Vision Language Models mit Encoder, Adapter und LLM	8
3.1	Überblick der Methodik	13
4.1	Ergebnisse der Information Extraction	27
4.2	Ergebnisse der Klassifikation	28
4.3	Confusion Matrices aller Modelle	29
4.4	Ergebnisse der Latenzmessung	30
4.5	Ergebnisse der Information Extraction	31
4.6	Ergebnisse der Klassifikation	32
4.7	Confusion Matrices aller Modelle	32
4.8	Ergebnisse der Latenzmessung pro Dokument	33
4.9	Ergebnisse der VRAM-Nutzung des geladenen Modells	34
4.10	Ergebnisse der VRAM-Nutzung während der Inferenz pro Dokument	35
4.11	Ergebnisse des Energieverbrauchs pro Dokument	36



## **Tabellenverzeichnis**

2.1 Benchmark-Ergebnisse der evaluierten Modelle[Bai 24, Qwen 25a, Agra 24] . . . . . 9



# **Listingverzeichnis**

1	Preprocessing der Dokumente . . . . .	14
2	JSON-Schema des Dokumententyps KG5b . . . . .	15
3	JSON-Schema des Dokumententyps Ausbildungsvertrag . . . . .	16
4	JSON-Schema des Dokumententyps Sonstiges . . . . .	16
5	Basisklasse der verschiedenen Comparator . . . . .	18
6	Comparator für Namen . . . . .	18
7	Konfiguration des KG5b-Formulars . . . . .	19
8	Ergebnis-Dictionary mit den Metadaten des gesamten Dokumentenkorpus . . . . .	20
9	Parameter, der bestimmt, ob Halluzinationen bestraft werden . . . . .	21
10	Finaler Prompt der Modellauswahl . . . . .	23
11	OpenAI Message Format . . . . .	24
12	Prompt für das Fine-Tuning des Qwen-2.5-VL-7B . . . . .	25
13	Konfiguration des Fine-Tunings . . . . .	25
14	Laden des Pixtral-Modells . . . . .	43
15	Laden der Qwen-Modelle . . . . .	44



## Literaturverzeichnis

- [Agra 24] P. Agrawal *et al.* “Pixtral 12B”. 2024.
- [Arbe 22] B. für Arbeit. “Erklärung zum Ausbildungsverhältnis (KG 5b)”. [https://www.arbeitsagentur.de/datei/dok\\_ba031910.pdf](https://www.arbeitsagentur.de/datei/dok_ba031910.pdf), 2022.
- [Arbe 24] B. für Arbeit. “Kindergeld / Kinderzuschlag Jahreszahlen 2024”. [https://statistik.arbeitsagentur.de/Statistikdaten/Detail/202412/famka/famka-jz/famka-jz-d-0-202412-pdf.pdf?\\_\\_blob=publicationFile&v=3](https://statistik.arbeitsagentur.de/Statistikdaten/Detail/202412/famka/famka-jz/famka-jz-d-0-202412-pdf.pdf?__blob=publicationFile&v=3), 2024.
- [Arbe 25] B. für Arbeit. “Auswirkungen des demografischen Wandels auf den Arbeitsmarkt”. [https://statistik.arbeitsagentur.de/DE/Statischer-Content/Statistiken/Themen-im-Fokus/Demografie/Generische-Publikationen/Bericht-Demografie.pdf?\\_\\_blob=publicationFile](https://statistik.arbeitsagentur.de/DE/Statischer-Content/Statistiken/Themen-im-Fokus/Demografie/Generische-Publikationen/Bericht-Demografie.pdf?__blob=publicationFile), 2025.
- [Bai 24] S. Bai *et al.* “Qwen2.5-VL Technical Report”. 2024.
- [Blec 23] L. Blecher *et al.* “Nougat: Neural Optical Understanding for Academic Documents”. 2023.
- [Cui 21] L. Cui *et al.* “Document AI: Benchmarks, Models and Applications”. 2021.
- [Hand 25] D. I. und Handelskammer. “Berufsausbildungsvertrag”. <https://www.dihk.de/resource/blob/140492/9f4fa2617d668aa62f5d3caad77cdbaa/bildung-musterausbildungsvertrag-data.pdf>, 2025.
- [Hu 21] E. J. Hu *et al.* “LoRA: Low-Rank Adaptation of Large Language Models”. 2021.
- [Jura 26] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Stanford University, 2026. Draft of Chapter 18: Information Extraction.
- [Kala 23] D. Kalajdzievski. “A Rank Stabilization Scaling Factor for Fine-Tuning with LoRA”. 2023.
- [Katt 18] A. R. Katti *et al.* “Chargrid: Towards Understanding 2D Documents”. 2018.
- [Kim 22] G. Kim *et al.* “OCR-free Document Understanding Transformer”. 2022.

- [Li 23] M. Li *et al.* “TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models”. 2023.
- [Li 25] Z. Li *et al.* “A Survey of State of the Art Large Vision Language Models: Alignment, Benchmark, Evaluations and Challenges”. 2025.
- [Liu 21] P. Liu *et al.* “Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing”. 2021.
- [Mang 22] S. Mangrulkar *et al.* “PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods”. <https://github.com/huggingface/peft>, 2022.
- [Mine 21] M. Minesh, K. Dimosthenis, and J. C.V. “DocVQA: A Dataset for VQA on Document Images”. 2021.
- [NVID 22] NVIDIA. “NVIDIA A40”. <https://www.nvidia.com/de-de/data-center/a40/>, 2022.
- [Pixt 24] Pixtral. “Pixtral-12B-2409”. <https://huggingface.co/mistralai/Pixtral-12B-2409>, 2024.
- [Qwen 25a] Qwen. “Qwen2.5VL-32B-instruct”. <https://huggingface.co/Qwen/Qwen2.5-VL-32B-Instruct>, 2025.
- [Qwen 25b] Qwen. “Qwen2.5VL-7B-instruct”. <https://huggingface.co/Qwen/Qwen2.5-VL-7B-Instruct>, 2025.
- [Wang 23] D. Wang *et al.* “DocLLM: A layout-aware generative language model for multimodal document understanding”. 2023.
- [Xu 20] Y. Xu *et al.* “LayoutLM: Pre-training of Text and Layout for Document Image Understanding”. 2020.
- [Yuji 07] L. Yujian and L. Bo. “A Normalized Levenshtein Distance Metric”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

# Glossar

**Accuracy** Metrik, die den Anteil der insgesamt korrekten Vorhersagen an der Gesamtzahl der getroffenen Vorhersagen beschreibt.. i

**F1-Score** Das harmonische Mittel aus Precision und Recall.. i

**Few-Shot Prompting** Ein Ansatz, bei dem dem Modell mehrere Beispiele im Prompt übergeben werden.. i

**FN** FN (False Negative) sind alle positiven Fälle, die fälschlicherweise als negativ klassifiziert wurden.. i, 20

**FP** FP (False Positive) sind alle negativen Fälle, die fälschlicherweise als positiv klassifiziert wurden.. i, 20

**GPU** Eine GPU (Graphics Processing Unit) ist ein spezialisierter Prozessor in einem Computer oder Server, der rechenintensive Arbeiten parallelisiert.. i, 39, 43

**IDE** Eine IDE (Integrated Development Environment) ist eine Entwicklungsumgebung die spezielle Werkzeuge für Entwickler bereitstellt.. i, 6

**Information Extraction** Information Extraction (IE) umfasst nach Jurafsky und Martin die Aufgabe, Ereignisse oder Situationen in Dokumenten zu identifizieren und die entsprechenden Felder eines vorgegebenen Templates zu füllen[Jura 26]. Ziel hierbei ist es, vordefinierte Entitäten aus einem Dokumentenbild zu extrahieren und diese in ein maschinenlesbares Schema zu überführen.. i, 10, 27, 28, 30, 31, 49

**JSON** JSON (JavaScript Object Notation) basiert auf Schlüssel-Wert-Paaren und ist ein textbasiertes Datenformat zur strukturierten Speicherung und Übertragung von Daten.. i, 11, 15, 16, 17, 19, 20, 22, 24, 38, 44, 45, 48

**KV-Cache** Der KV-Cache (Key-Value-Cache) speichert die Key-Value-Vektoren der bereits generierten Token, sodass nur die Vektoren der neuen Token berechnet werden müssen.. i

**Latenz** Die Zeitdauer, die das System für die Inferenz einer einzigen Antwort benötigt.. i, 11, 13, 21, 30, 33, 39, 40, 41, 42, 47

**Levensthein-Similarity** Die Levenshtein-Similarity ist ein aus der Levenshtein-Distanz abgeleiteter, normalisierter Ähnlichkeitswert zwischen zwei Zeichenketten. Formal basiert sie auf der Levenshtein-Distanz, also der minimalen Anzahl von Einfügungen, Löschungen oder Ersetzungen, die nötig sind, um eine Zeichenkette in eine andere zu überführen[Yuji 07].. i

**logprobs** Logprobs (Logarithmic Probabilities) sind die logarithmierten Wahrscheinlichkeiten der möglichen Token pro Inferenzschritt.. i, 44

**OCR** OCR (Optical Character Recognition) ist ein Verfahren zu Texterkennung, das hand- und maschinenschriftliche Zeichen aus Bildern extrahiert und in einen Text umwandelt. . i, 1, 6, 10, 11, 12, 37, 38

**One-Shot Prompting** Eine Technik, bei der dem Modell genau ein Beispiel für die Lösung im Prompt bereitgestellt wird.. i

**OpenAI Message Format** Das OpenAI Message Format ist ein standardisiertes, rollenbasiertes Datenformat zur Interaktion mit Modellen, bei dem Nachrichten nach Rollen unterteilt und Inhalte als Array aus verschiedenen Typen wie Text und Bild definiert werden.. i, 24

**Precision** Gibt den Anteil der tatsächlich positiven Fälle an allen vom Modell als positiv klassifizierten Fällen an.. i

**Prompt Engineering** Ist ein Prozess, um die effektivste Anweisung für eine spezifische Aufgabe zu finden[Liu 21].. i

**R-K-F-Formel** Strategie im Prompt Engineering zur strukturierten Anweisung eines LLMs. Dabei steht R für Rolle (Rolle, die das Modell einnehmen soll), K für Kontext (relevante Hintergrundinformationen) und F für Format (Vorgabe der gewünschten Ausgabestruktur).. i

**Recall** Gibt den Anteil der korrekt als positiv erkannten Fälle an allen tatsächlich vorhandenen positiven Fällen an.. i

**TN** TN (True Negative) sind alle negativen Fälle, die tatsächlich als negativ klassifiziert wurden. . i, 21

**TP** TP (True Positive) sind alle positiven Fälle, die tatsächlich als positiv klassifiziert wurden. . i, 20

**Unsloth** Unsloth ist ein Open-Source-Framework zum beschleunigten und speichereffizienten Fine-Tuning von Large Models, das speziell für Methoden des PEFT optimiert ist.. i, 24

**vLLM** vLLM ist eine Open-Source-Engine für die effiziente Inferenz von großen Sprachmodellen.. i, 44, 48

**VQA** VQA (Visual Question Answering) ist eine Aufgabe, bei der ein Modell ein Bild analysiert und auf Basis dessen eine Frage beantwortet.. i, 9

**VRAM** VRAM (Video Random Access Memory) ist ein spezieller Grafikspeicher auf einer GPU, der zur schnellen Speicherung und Verarbeitung während rechenintensiver Anwendungen verwendet wird. . i, 6, 8, 13, 21, 24, 39, 42, 43, 44

**VRAM-Auslastung** Der maximale Bedarf an Grafikspeicher der GPU, der während der Ausführung eines Modells gemessen wird.. i

**YOLO** Ein YOLO-Modell (You Only Look Once) ist ein Deep-Learning-Verfahren zur Objekterkennung, das Objekte und die dazugehörigen Klassen in einem einzigen Durchlauf erkennt. Für die Detektion werden Bounding-Boxes und Klassenwahrscheinlichkeiten berechnet, was eine schnelle Erkennung ermöglicht. . i, 1, 6, 40, 41

**Zero-Shot Prompting** Eine Methode, bei der dem Modell eine Aufgabe gestellt wird, ohne dass zusätzliche Beispiele im Prompt enthalten sind.. i

Hinweis: Diese Erklärung ist mit Originalunterschrift (nicht gescannt) in das Papierexemplar der Abschlussarbeit fest einzubinden. Eine Spiralbindung ist nicht zulässig. Das digitale Exemplar enthält einen Scan der Erklärung mit Unterschrift.

### Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Müller Vorname: Lukas Matrikel-Nr.: 3698673

Fakultät: Informatik Studiengang: Informatik

Semester: 7

#### Titel der Abschlussarbeit:

Vergleichsanalyse von multimodalen Large Language Models und einer OCR/YOLO-Pipeline zur Dokumentenklassifikation für die Bundesagentur für Arbeit

Ich versichere, dass ich die Arbeit selbstständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe. Außerdem versichere ich, dass der Hauptteil des Papierexemplares und des digitalen Exemplares identisch sind. Das digitale Exemplar enthält, falls gefordert, lediglich weitere Anlagen.

Nürnberg, 25.02.2026,

Ort, Datum, Unterschrift Studierende/Studierender

### Erklärung der/des Studierenden zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit  genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,  
 genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von 0 Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format an die Betreuer übermittelt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Nürnberg, 25.02.2026,

Ort, Datum, Unterschrift Studierende/Studierender

**Datenschutz:** Die Antragstellung ist regelmäßig mit der Speicherung und Verarbeitung der von Ihnen mitgeteilten Daten durch die Technische Hochschule Nürnberg Georg Simon Ohm verbunden. Weitere Informationen zum Umgang der Technischen Hochschule Nürnberg mit Ihren personenbezogenen Daten sind unter nachfolgendem Link abrufbar: <https://www.th-nuernberg.de/datenschutz/>